FPGA Implementation of Data Encryption and Decryption using optimized LED algorithm

[1]Sunita Bangari, [2]E. Elavarasi

Dept. Of ECE, AMCEC, Prof. ECE Dept. AMCEC, BANGALORE
Email: [1]sunitabangari@gmail.com, [2]elu_ye@yahoo.co.in

Abstract— This paper presents light weight encryption device (LED) on field programmable gate array (FPGA). An optimized code for the light weight encryption algorithm with 128 bits key size and 64 bits block size has been developed. We have designed iterative architecture for speed improvements and hardware elements can be reused for every rounds of operation. A light weight encryption algorithm is essential for secure communication. Further we are introducing key scheduling for reducing the number of rounds. The target device is Spartan-3 FPGA.

Index Terms- LED (light weight encryption device), FPGA, Key scheduling.

I. INTRODUCTION

Recently, due to the advent of resource-constrained trends, such as smart phones and smart devices, the computing environment is changing. Because our daily life is deeply intertwined with ubiquitous networks, the importance of security is growing. A lightweight encryption algorithm is essential for secure communication between these kinds of resource-constrained devices, and many researchers have been investigating this field.

The large deployment of these lightweight services resulted in security and privacy problems. In order to make them secure, special cryptographic techniques should be applied as they have limited resources and require low power consumption. Therefore, lightweight cryptography is used for securing these constrained devices [1].

Techniques for securing resource-constrained devices such as RFID (Radio-frequency Identification) tags have been proposed. In 2005, Lim and Korkishko [2] presented a lightweight block cipher called mCrypton that encrypts plaintext into cipher text by using 4 by 4 nibble (4-bit) matrix-based simple operations such as substitution (S-Box), permutation, transposition, and key addition (XOR). The following year, Hong et al. [3] proposed a lightweight block cipher called HIGHT, which has a Feistel structure and operates with simple calculations such as XOR, addition, subtraction, and rotation. In 2007, Bogdanov et al. [4] introduced PRESENT, which is comprised of substitution, permutation, and XOR. In 2009, KATAN and KTANTAN were proposed by Cammoere et al. [5] KATAN divides plaintext into two parts and stores them into two registers, and the outputs from non-linear functions are stored in the least significant bit (LSB) of each other's register. On the other hand KTANTAN is a fixed-key version of KATAN and has a different key scheduling scheme. In the same year, Rotor-based Humming Bird was proposed by Revere Security. However, these algorithms have been revealed to be vulnerable to chosen-IV attacks and chosen message attacks. Two years later, HummingBird2 [6], an improved version of Humming Bird, was proposed. In 2011, Guo et al. Proposed a lightweight cipher LED.

In cryptography, encryption is the process of encoding messages (or information) in such a way that third parties cannot read it, but only authorized parties can. Encryption doesn't prevent hacking but it prevents the hacker from reading the data that is encrypted. In an encryption scheme, the message or information (referred to as plaintext) is encrypted using an encryption algorithm, turning it into an unreadable cipher text. This is usually done with the use of an encryption key, which specifies how the message is to be encoded. Any adversary that can see the cipher text should not be able to determine anything about the original message. An authorized party, however, is able to decode the cipher text using a decryption algorithm, that usually requires a secret decryption key that adversaries do not have access to. For technical reasons, an encryption scheme usually needs a key-generation algorithm to randomly produce keys. Usually, small chip size and reasonably fast encryption is preferred for cryptographic hardware for small devices in resource constrained environments such as RFID tags or smart meters for smart grides.

This paper is organized as follows. Section II will gives a design approach and specifications of LED. Section III will discuss the LED rounds of functions and section IV

will provide a Iterative architecture. Section V shows simulation results. Section VI concludes the paper.

II. DESIGN APPROACH AND SPECIFICATIONS OF LED

The design of LED will inevitably have many parallels with this established approach, and features such as S boxes, Shift Rows, and (a variant of) Mix Columns will all take their familiar roles. The LED cipher is 64 bit block cipher with two primary instances taking 64-bit and 128-bit keys. The cipher state is conceptually arranged in a (4×4) grid where each nibble represents an element from GF (2^4) with the underlying polynomial for field multiplication given by

$X^4 + X + 1$.

The LED encryption function is as shown in fig 1. It has 2 keys (K1 & K2) for every step and 48 rounds of operation. Initially plain text (P) is EX-OR'ed with K1, where K1 ranges from 0 to 63 bits. Later it will perform 4 rounds of operations. After the completion of first four rounds keys (K2) is EX-OR'ed, where K2 ranges from 64 to 127 bits.

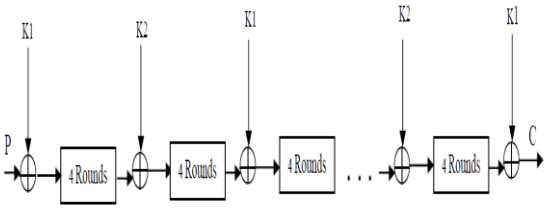

Fig.1 LED encryption function

The single step performs 4 rounds of operations is as shown in fig 2. For a 64-bit plaintext m the 16 four-bit nibbles m0||m1||….. ||m14||m15 are arranged (conceptually) in a square array:

$$\begin{bmatrix} m_0 & m_1 & m_2 & m_3 \\ m_4 & m_5 & m_6 & m_7 \\ m_8 & m_9 & m_{10} & m_{11} \\ m_{12} & m_{13} & m_{14} & m_{15} \end{bmatrix}$$

LED operates on a state that consists of 16 4-bit cells. The 16 4-bit cells are arranged in 4 columns and 4 rows. This only matters for the description of the round function since parts of it operate on single cells, rows, and columns of the state.

The LED round function has many similarities to that of the AES block cipher. It consists of four parts Add Constants, Sub Cells, Shift Rows, and Mix Columns.

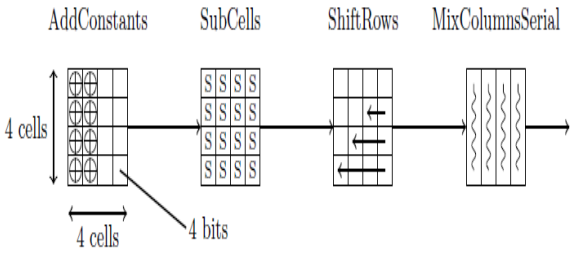

Fig. 2. LED round function

III. LED ROUND FUNCTION

A. Add Constants

The Add Constants part of the round function serves to prevent slide attacks that would threaten the cipher due to its very simple key schedule. At the begin of every round different constants are being XOR'ed into the state [7].

The constants follow this scheme:

$$\begin{vmatrix} 0 & (rc_5||rc_4||rc_3) & 0 & 0 \\ 1 & (rc_2||rc_1||rc_0) & 0 & 0 \\ 2 & (rc_5||rc_4||rc_3) & 0 & 0 \\ 3 & (rc_2||rc_1||rc_0) & 0 & 0 \end{vmatrix}$$

with rc being a six bit wide linear feedback shift register with rc5 ⊕ rc4 ⊕ 1 as its feedback function. The initial value of the rc is all six bits zero. The values of rc are:

Table 1 Constants of each round

round	0	1	2	3	4	5	6	7	8	9	10	11
rc	01	03	07	0f	1f	3e	3d	3b	37	2f	1e	3c
round	12	13	14	15	16	17	18	19	20	21	22	23
rc	39	33	27	0e	1d	3a	35	2b	16	2c	18	30
round	24	25	26	27	28	29	30	31	32	33	34	35
rc	21	02	05	0b	17	2e	1c	38	31	23	06	0d
round	36	37	38	39	40	41	42	43	44	45	46	47
rc	1b	36	2d	1a	34	29	12	24	08	11	22	04

B. Sub cells

The S-Box is identical to the one being used in PRESENT. It also is the only source of non-linearity in the cipher so that without it one could easily break LED using Gaussian elimination.

LED cipher re-uses the present S_box which has been adopted in many lightweight cryptographic algorithms.

The action of this box in hexadecimal notation is given by the following table.

Table 2 S_BOX of LED

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2

C. Shift Rows

Shift Rows operates on the rows of the state and works exactly like its counterpart in AES. Every row is rotated left by its index minus one.

$$\begin{vmatrix} s_1 & s_2 & s_3 & s_4 \\ s_5 & s_6 & s_7 & s_8 \\ s_9 & s_{10} & s_{11} & s_{12} \\ s_{13} & s_{14} & s_{15} & s_{16} \end{vmatrix} \rightarrow \begin{vmatrix} s_1 & s_2 & s_3 & s_4 \\ s_6 & s_7 & s_8 & s_5 \\ s_{11} & s_{12} & s_9 & s_{10} \\ s_{16} & s_{13} & s_{14} & s_{15} \end{vmatrix}$$

Fig.3 shift rows operations of LED algorithm

Even though this part of the round function is very simple, it becomes a powerful diffusion layer for the cipher in combination with Mix Columns.

D. Mix Columns

Just like AES LED uses a maximum distance separable code to transform the columns of its state. And just as in AES [8] this step is called "Mix Columns". This step consists of multiplying every column of the state with a fixed matrix that has the maximum distance sparse property. LED uses a matrix A over the extension field (GF) (2) $[X]/(X^4 + X + 1)$:

$$A = \begin{pmatrix} 4 & 1 & 2 & 2 \\ 8 & 6 & 5 & 6 \\ 11 & 14 & 10 & 9 \\ 2 & 2 & 15 & 11 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 4 & 1 & 2 & 2 \end{pmatrix}^4$$

While the Sub Cells part of the round function delivers all the non-linearity (confusion) for the cipher, diffusion is achieved by Mix Columns and Shift Rows; every cell in the output column depends on every cell in the input column [9].

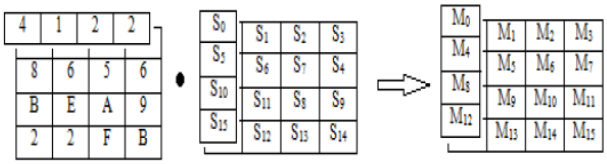

Fig. 4 Mix- columns operations of LED algorithm

The following equations represent the Galois Field Multiplication in (2^4)

M0= (4×S0) XOR (1×S5) XOR (2×S10) XOR (2×S15)

M1= (8×S0) XOR (6×S5) XOR (5×S10) XOR (6×S15)

M2= (B×S0) XOR (E×S5) XOR (A×S10) XOR (9×S15)

M3= (2×S0) XOR (2×S5) XOR (F×S10) XOR (B×S15)

The Decryption method of LED algorithm will perform inverse mix-column operation.

IV. ITERATIVE ARCHITECTURE

In this work, we have designed iterative architecture for speed improvements and introducing T_BOX for reducing computations. T_BOX is designed by combining S_BOX and Mix columns serial step. 128 bit key size is used and it has 48 rounds of operation. By using T_BOX we can achieve more speed compared to LED encryption using Galois field multiplier.

The main focus of this work is speed improvements and area reduction. In this iterative architecture hardware elements can be reused for every rounds of operation and we can achieve reduction in hardware utilization. In the first clock cycle, input block of data is fed to the circuit through the multiplexer. In each subsequent cycle, one round of cipher is evaluated and the result is fed back to the circuit through the multiplexer.

Table 3 shows the constants of T_BOX (transformation box). mix columns serial step requires more computations for this reason we have derived T_BOX (transformation box).

Table 3 Transformation Box

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	C	5	6	B	9	0	A	D	3	E	F	8	4	7	1	2
2	B	A	C	5	1	0	7	9	6	F	D	3	8	E	2	4
3	0	F	A	E	8	0	D	0	0	1	2	B	C	0	3	6
4	5	7	B	A	2	0	E	1	C	D	9	6	3	F	4	8
5	9	2	D	1	B	0	4	C	F	3	C	E	7	8	5	A
6	E	D	7	F	3	0	9	8	A	2	4	5	B	1	6	C
7	0	8	1	4	A	0	3	0	0	C	B	D	F	0	7	E
8	A	E	5	7	4	0	F	2	B	9	1	C	6	D	8	3
9	6	B	3	C	D	0	5	F	8	7	E	4	2	A	9	1
A	1	4	9	2	5	0	8	B	D	6	C	F	E	3	A	7
B	D	1	F	9	C	0	2	6	E	8	3	7	A	4	B	5
C	0	9	E	D	6	0	6	0	0	4	8	A	5	0	C	B
D	0	C	8	6	F	0	6	0	0	A	7	2	1	0	D	9
E	4	3	2	8	7	0	6	A	1	B	5	9	D	C	E	F
F	8	6	4	3	E	0	C	7	2	5	A	1	9	B	F	D

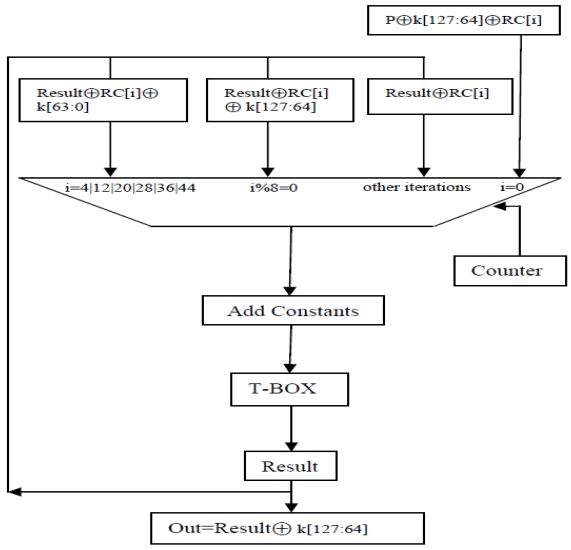

Fig 5 iterative architecture

For the first round, plain text is performing EX-OR operation with the key size of 64 to 127 bits and also will perform EX-OR operation with the constant RC[i] to get the next round's Add Constant (AC). For the rounds where i=4|12|20|28|36|44, the previous round result is EX-OR'ed with 0 to 63 bits of key and also with corresponding RC[i] to get next round's AC. For the rounds where (i mod 8) = 0, the previous round result is EX-OR'ed with 64 to 127 bits of key and also with corresponding RC[i]. For the other rounds of iteration the result of the previous round is EX-OR'ed with the corresponding RC[i] to obtain AC of the next immediate round.

This AC output is given to the input of T-Box and the corresponding result from T-Box is fed to the multiplexer. At the end of the final round (i=47) result is EX-OR'ed with 64 to 127 bits of key to get final encrypted output (cipher text).

V. RESULTS AND DISCUSSION

The encryption LED module design is targeted to Spartan 3 FPGA. The design is synthesized using Xilinx XST synthesis tool. The main focus of this work is area reduction and speed improvements. Table 4 shows the device utilization of LED encryption module. we achieved better results compared to the existing method of LED encryption module. The encryption module is simulated with a 64 bit input plain text and 128 bit key.

Plain text: 0123456789ABCDEF

Key: 0123456789ABCDEF0123456789ABCDEF

After 48 rounds of operation the cipher text obtained is,

Cipher text: 3131C231205C3664

Table 4. Device utilization

Logic utilization	Previous work	Proposed work
Number of slices	204 out of 33280	194 out of 33280
Number of 4 input LUTs	358 out of 66560	319 out of 66560
Number of bonded IOBs	257 out of 633	258 out of 633
Number of slice flip flops	6 out of 66560	231 out of 66560
Number of BRAMs	32 out of 104	16 out of 104
Number of GCLKs	1 out of 8	1 out of 8
Maximum frequency	93.552 MH_z	130.314 MH_z

VI. CONCLUSION

In this paper, we have presented LED block cipher on FPGA. We have designed iterative architecture for speed improvements and hardware elements can be reused for every rounds of operation. The T_box (transformation box) has been derived by combining both S_BOX and Mix- column serial step for reducing computations. The targeted design is Spartan 3 FPGA. The design is synthesized using Xilinx XST synthesis tool.

REFERENCES

[1] Banraplang Jyrwa, Roy Paily, An Area-Throughput Efficient FPGA Implementation of Block Cipher AES Algorithm, International Conference on Advances in computing, control, and telecommunication technologies 2009.

[2] Lim, C.H., Korkishko, T.: mCrypton – A Lightweight Block Cipher for Security of Low-Cost RFID Tags and Sensors. In: Song, J., Kwon, T., Yung, M. (eds.) WISA 2005. LNCS, vol. 3786, pp. 243–258. Springer, Heidelberg (2006)

[3] Hong, D., Sung, J., Hong, S., Lim, J., Lee, S., Koo, B., Lee, C., Chang, D., Lee, J., Jeong, K., Kim, H., Kim, J., Chee, S.: HIGHT: A New Block Cipher Suitable for Low-Resource Device. In: Goubin, L.,Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 46–59. Springer, Heidelberg (2006)

[4] Bogdanov. A, Knudsen L.R, Leander. G, Paar. C, Poschmann. A, Robshaw M.J.B, Seurin. Y, and Vikkelsoe. C, "PRESENT: An Ultra-Lightweight Block Cipher" Cryptographic Hardware and Embedded Systems Springer-Verlag, 2007, LNCS 4727, pp. 450-466.

[5] C. DE CANNIÈRE, O. DUNKELMAN AND M. KNEŻEVI´C . "KATAN & KTANTAN – A Family of Small and Efficient Hardware-Oriented Block Ciphers." CHES 2009, LNCS 5747, pp. 272–288. Springer (2009)

[6] X. FAN, H. HU, G. GONG, E. M. SMITH AND D. ENGELS. "Lightweight Implementation of Hummingbird Cryptographic Algorithm on 4-Bit Microcontroller." The 1st International Workshop on RFID Security and Cryptography 2009 (RISC'09), pp. 838–84. Springer (2009).

[7] A. Biryukov and D. Wagner. Slide Attacks. In Fast Software Encryption, pages 245–259. Springer, 1999.

[8] F.P.U.B. NIST. 197, Advanced Encryption Standard (AES), November 2001.

[9] Axel York Poschmann, "Lightweight Cryptography: Cryptographic Engineering for a pervasive world", 2009.

[10 Cetin Kaya Koc, "Cryptographic Engineering", Springer 2009.

[11] Thomas Eisenbarth, ChristofPaar, Axel Poschmann, Sandeep Kumar and Leif Uhsadel, "A Survey of Lightweight- Cryptography Implementations" proc. IEEE design and test of ICs for Secure Embedded Computing,2007.

◈◈◈