FPGA Implementation of AES Algorithm using Composite Field Arithmetic

N.Anitha Christy and P.Karthigaikumar
Department of Electronics and Communication Engineering,
Karunya University, India.

***Abstract*- A Low area Advanced Encryption Standard (AES)-128 bit algorithm is proposed in this paper. This technique is implemented using Composite Field Arithmetic (CFA) in byte substitution block, inverse byte substitution block and key expansion block of AES algorithm. The Composite field arithmetic technique provides a low area than the Look Up Table (LUT) in S-Box/Inverse S-Box. The proposed technique is presented with multistage sub-pipelined architecture in order to increase the throughput and its performance is compared with the previous FPGA implementations.**

Index Terms- AES, Composite Field Arithmetic, Field Programmable Gate Array.

I. INTRODUCTION

Network Security is important in all aspects of life. This provides security for the data being transmitted from one point to another point. Cryptography is a form of security in which the input data is converted to encrypted data and is transmitted in the Encryption module and in the decryption module; the encrypted data is converted again to decrypted data which is same as the input data. Several cryptographic algorithms have been proposed in the past few years. Some of the cryptographic algorithms are Blow fish, DES, Triple DES, SAFER, IDEA, RC4, etc.

The Advanced Encryption Standard (AES) algorithm was selected as the winner algorithm by NIST [1] (National Institute of Standards and Technology), which is the federal standard to protect the sensitive information. AES has already received widespread use because of its high security, high performance in both hardware and software implementations.

AES is a 128 symmetric data block cipher with 128,192 or 256 bits key. The data block is described in a 4x4 array known as state array [2]. The data block is sent through four basic functions: Substitute bytes, Shift Rows, Mix Column and Add Round Key. These four steps make one round of the AES. The number of rounds depends upon the Key length (Nk) words. The key length (Nk), Block Size (Nb) and the number of rounds (Nr) combination for AES-128, AES-192 and AES-256 can be seen from the Table 1.

TABLE I

KEY-BLOCK-ROUND COMBINATION

	Key length (Nk words)	Block Size (Nb words)	Number of Rounds (Nr)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

The Mix Column round is excluded for the last round. The decryption is the reverse order of the ciphering process. Operations are just similar and inverse of the encryption process.

Many implementations are done in software but it seems to be too slow for fast applications such as routers and wireless communication systems [3]. The implementations are physically secure since attacking from outside is very difficult.

The large area implementation of AES architectures may not be suitable for some low end embedded applications. Hence, reducing the hardware resources to gain a compact and efficient implementation circuit is an increasing demand [4].

The rest of this paper is organized as follows: section 2 is about the AES architecture. Section 3 describes the proposed technique Composite Field Arithmetic and its

978-1-4577-1546-4/12/$26.00 ©2012 IEEE

steps to construct the S-Box/Inverse S-Box. Section 4 briefs the implementation results and the last section is conclusion and future work.

II. AES Architecture

The 128 bit data is divided into 16 bytes. These bytes are mapped to a 4x4 array called the State. This State undergoes four different functions. They are

A. Sub Bytes

Substitute byte is a non-linear byte substitution from the S-Box. It is done by using two methods [5]. One is by using Look Up Table and the other is by using Combinational Logic. The values of the S-Box are obtained by taking multiplicative inverse over $GF(2^8)$ and affine transformation.

B. Shift Rows

In Shift Rows Phase [6], the rows of the state are shifted over different offsets. In AES-128 algorithm, the first row remains the same and the second row is shifted to the left once. Similarly third and fourth row are shifted to the left cyclically by three and four shifts.

C. Mix Column

In Mix Column Phase, each column of the state is multiplied with a constant polynomial over a finite field [7]. For the 128 bit key, each column is multiplied by the known matrix (1) which is given by,

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \quad (1)$$

In this matrix, multiplication by 1 means leaving unchanged, multiplication by 2 means shifting byte to the left and multiplication by 3 means shifting to the left and then performing ex-or with the initial unshifted value.

D. Add Round Key

In the Add Round Key phase [8], the data is combined with the key using the ex-or operation. The key is obtained from the Rijndael's Key Schedule.

E. Key Scheduling

In Key scheduling unit [9], round keys are generated beforehand and stored in memory or generated on the fly. In the former, the round keys can be read out from memory by using corresponding addresses and there is no extra delay. In the latter, the round keys are generated on the fly. In this paper, the round keys are generated on the fly in order to reduce the area. However, this approach is suitable for the applications where the key changes constantly.

III. Composite Field Arithmetic

The byte substitution in AES algorithm is obtained by two steps [10]. First, finding the multiplicative inverse of the number and then applying the affine transformation. In AES-128 algorithm, a data block of 16 bytes is sent where each byte is applied for byte substitution. It is complex to calculate the multiplicative inverse in $GF(2^8)$. Hence, a mapping of lower order Galois field to higher order Galois field can be accomplished using the following polynomials.

$$GF(2) \longrightarrow GF(2^2) \quad :X^2+X+1 \quad (2)$$

$$GF(2^2) \longrightarrow GF((2)^2) \quad :X2+X+\phi \quad (3)$$

$$GF((2^2)^2) \longrightarrow GF(((2^2)^2)^2) \quad :X2+X+\lambda \quad (4)$$

The constants ϕ (3) and λ (4) are chosen to keep the function irreducible, where the value of ϕ is taken as 10 and the value of λ is taken as 1100. The Sub Bytes using Composite Field Arithmetic is shown in the Fig.1.

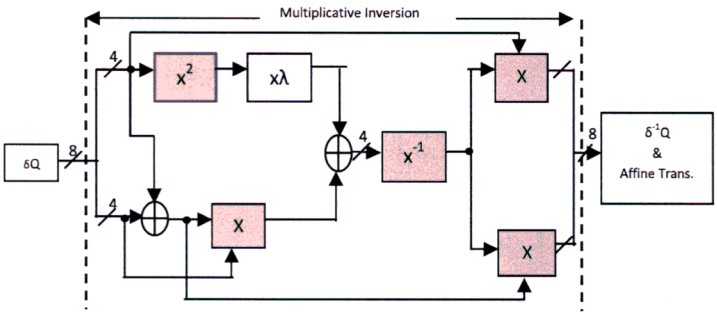

Figure 1.SubBytes using Composite Field Arithmetic Technique

A. Isomorphic Mapping

An isomorphic mapping function $f(q)=\delta \times Q$ should be applied to map the representation of an element in $GF(2^8)$ to its composite field [11]. The field elements of $GF(2^8)$ are mapped to elements in some lower order isomorphic composite fields. The isomorphic mapping

function (δ) is decided by the field polynomials of GF (2^8) and its composite fields. The δ matrix is shown in (5).

$$\delta \times Q = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \\ q_4 \\ q_5 \\ q_6 \\ q_7 \end{bmatrix} \quad (5)$$

B. Multiplicative Inverse

The Multiplicative Inverse in GF (2^4) can be calculated by using the (6).

$$(bx+c)^{-1} = b(b^2\lambda + c(b+c))^{-1}x + (c+b)(b^2\lambda + c(b+c))^{-1} \quad (6)$$

Multiplicative Inverse [12] consists of multiplier in $F(2^4)$,multiplier in $GF(2^2)$,squarer in $GF(2^4)$,constant multipliers ($x\lambda$ and $x\phi$) and inversion in $GF(2^4)$ which is shown in the Fig.2.

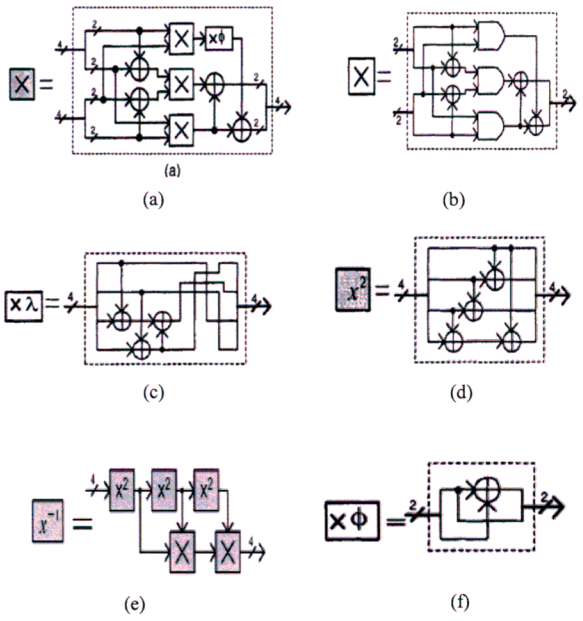

Figure 2.Sub Blocks of Multiplicative Inverse (a)Multiplier in $GF(2^4)$; (b) Multiplier in $GF(2^2)$; (c) Constant Multiplier ($x\lambda$);(d) Squarer in $GF(2^4)$; (e) Inverse in $GF(2^4)$; (f) Constant Multiplier ($X\varphi$)

C. Inverse Isomorphic mapping and Affine transform

The computation result is mapped back to the original field by using the reverse function δ^{-1} which is shown in (7).

$$\delta^{-1} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \quad (7)$$

Affine transformation is done by multiplication by a matrix followed by addition of a vector [13].
The affine transformation is defined in (8).

$$(A^4 + A^3 + A^2 + A + I)X + 63 \quad (8)$$

The equivalent representation of (8) is given in (9).

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \quad (9)$$

where A is the matrix which is given in (10)

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad (10)$$

The inverse S-box is simply the S-box run in reverse. For example, the inverse S-box of 0xdb is 0x9f. It is calculated by first calculating the inverse affine transformation of the input value, followed by the multiplicative inverse and isomorphic mapping.

The inverse affine transformation is as follows in (11).

$$\begin{bmatrix}0&0&1&0&0&1&0&1\\1&0&0&1&0&0&1&0\\0&1&0&0&1&0&0&1\\1&0&1&0&0&1&0&0\\0&1&0&1&0&0&1&0\\0&0&1&0&1&0&0&1\\1&0&0&1&0&1&0&0\\0&1&0&0&1&0&1&0\end{bmatrix}\begin{bmatrix}x_0\\x_1\\x_2\\x_3\\x_4\\x_5\\x_6\\x_7\end{bmatrix}+\begin{bmatrix}1\\0\\1\\0\\0\\0\\0\\0\end{bmatrix} \quad (11)$$

IV. Implementation Results And Discussion

The AES algorithm with Composite Field Arithmetic is implemented using FPGA devices. Xilinx 12.2 ISE is used to synthesize the design and provides the synthesis report. The code is pure VHDL that could easily be implemented on targeted FPGA devices, without changing the design. The Throughput [14] is obtained by calculating the formula.

$$\text{Throughput} = \frac{128 \text{ x Blocks per Cycle}}{\text{Area}} \quad (12)$$

128 is the block size in (12). The area occupied by the Look Up Table (LUT) and Composite Field Arithmetic(CFA) are compared and its throughput is calculated.

A. *Simulation in the Modelsim XE III 6.3c*

The basic AES is programmed in XILINX Virtex 6 (XC6VLX75T) device and simulated in MODELSIM XE III 6.3c. The Simulated waveform of the Composite field Arithmetic with the basic AES is shown in the Fig.3. The Composite Field Arithmetic is used in the Sub-Bytes, Inverse Sub-Bytes and in Key Scheduling unit. The encrypted and decrypted results are shown as hexadecimal in the waveform.

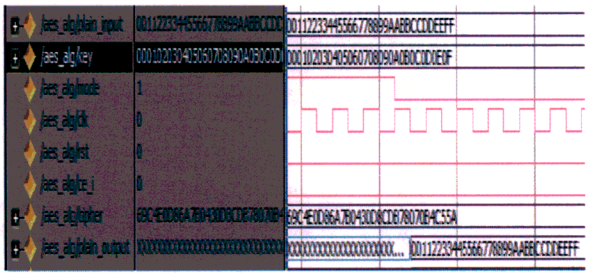

Figure 3.Simulation Waveform of the Basic AES with CFA

From the Fig.4, it is observed that the decrypted data is obtained and is same as that of the Plain Text. The results are obtained for different type of device and are compared with the relevant works in the AES field.

B. *Performance Comparison with the related works*

Several authors have done their research in AES algorithm. It has been impossible to find other FPGA-based AES implementations using exactly the same type of device. Hence, we have compared the obtained results with few recent papers. Table 2 shows the Performance Comparison of FPGA results with the existing works.

TABLE II

Perfomance Comparison Of Fpga Results With The Existing Works

Device	Author	Archi (S Box)	Area (slices)	Throughput (Gbps)
XC5VLX110T	[15]	LUT	4611	13.238
	This Work	**CFA**	**1156**	**9.120**
XC4VLX40	[16]	LUT	1725	0.497
	This Work	**CFA**	**1056**	**0.280**
XC3S1000	[17]	LUT	7606	2.19
	This Work	**CFA**	**2564**	**1.90**
XC6VLX75T	**This Work**	**LUT**	**1380**	**35.995**
		CFA	**1025**	**22.743**

From the Table 2, it is observed that the Area occupied by Composite Field Arithmetic (CFA) is less than the area occupied by the Look Up Table (LUT). Since in the Look Up Table, the values are pre-calculated using multiplicative inverse and affine transformation and uses memory to store the results. But, in CFA the values are calculated on the fly and they are not stored in separate memory and the values are used directly. Hence the area occupied is large in Look Up Table than CFA which is shown as the bar chart in the Fig.4.

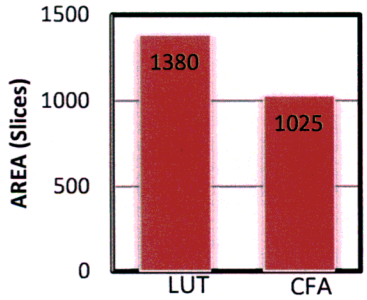

Figure 4.Bar Chart Representation of Area comparison of LUT and CFA

It is observed from the table 2 that the throughput decreases for CFA because as the values are calculated on the fly, it needs some time to compute. Hence the throughput of CFA is less than the Look Up Table.

V. Conclusion and Future Work

In this paper, an efficient architecture of the AES algorithm is presented in order to reduce the area. The Composite Field Arithmetic (CFA) occupies area which is much less than the area occupied by the Look Up Table (LUT).

The throughput can be increased by pipelining and sub-pipelining the Composite Field Arithmetic (CFA). The AES architecture can be further optimised by excluding the Shift Row operation. This will reduce the area to a great extent and will enhance the throughput.

Acknowledgment

The authors would like to acknowledge the management of Karunya University for providing us with all the tools and necessary support to carryout the research successfully. Also would like to thank Ann Caroline Jenifer of Karunya University for her help and support towards this paper.

References

[1] National Inst. of Standards and Technology, "Federal Information Processing Standard Publication 197, the Advanced Encryption Standard(AES),"Nov.2001.

[2] Chang.C.J, Hu.C.W, Chang.K.H, Cheng Chen.Y.C and Hsieh.C.C, "High Throughput32-bit AES Implementation in FPGA",pp 1806-1809,2008.

[3] Samiee.H,Atani.R.E,"A Novel Area-Throughput Optimizes Architecture for the AES Algorithm",International Conference on Electronic Devices,Systems and Applications,pp 29-32,2011.

[4] Luo.A.W, Qing Ming Yi and Min Shi, "Design and Implementation of Area-optimized AES based on FPGA",pp 743-746,2011.

[5] Yoo.S.M, Kotturi.D, Pan.D.W and Blizzard.J, "An AES crypto chip using a high speed parallel pipelined architecture",Science direct,Microprocessors and Microsystems,Vol.29,pp 317-326,2005.

[6] Fayed.M,Watheq El-Kharashi.M and Gebali.F, "A high speed,Fully Pipelined VLSI Architecture for real time AES",4th International Conference on Information and Communications technology(ICICT),pp 1-2,Dec 2006.

[7] Zhang.Y and Wang.X, "Pipelined Implementation of AES Encryption based on FPGA",pp 170-173,Dec. 2010.

[8] Thongkhome.K, Thanavijitpun.C and Choomchuay.S, "A FPGA Design of AES core architecture for portable hard disk",8th International Conference on Computer Science and Software Engineering(JCSSE),pp 223-228,2011.

[9] Qu.S,Shou.G,Hu.Y,Guo.Z and Qian.Z, "High Throughput, Pipelined Implementation of AES on FPGA", International Symposium on Information Engineering and Electronic Commerce, Ternopil, Ukraine,pp 542-550,May 2009.

[10] Chen.D, Shou.G, Hu.Y and Guo.Z, "Efficient Architecture and Implementations of AES",3rd International Conference on Advanced Computer Theory and Engineering(ICACTE),Vol.6,pp 295-298, Aug.2010.

[11] M.Masoumi and S.Mohammadi, "A New and Efficient Approach to protect AES against Differential Power Analysis", IEEE, pp 59-66,2011.

[12] Zhang.X and Parhi.K.K, "High speed VLSI Architectures for the AES Algorithm",IEEE Transactions on Very Large Scale Integration (VLSI)systems,Vol.12,No.9,pp 957-967,September 2004.

[13] Hammad.I,El-Sankary.K and Ezz El-Masry, "High Speed AES Encryptor with Efficient Merging Techniques",IEEE Embedded Systems Letters,Vol.2,No.3,pp 67-71,September 2010.

[14] Granado Criado .J.M, Vega Rodriguez.M.A,Sanchez Perez.J.M and Gomez Pulido.J.A, "A new methodology to implement the AES algorithm using partial and dynamic reconfiguration", INTEGRATION,the VLSI journal, Vol. 43,pp 72-80,2010.

[15] Reddy.S.K, Sakthivel.R and Praneeth.P, "VLSI Implementation of AES Crypto Processor for High Throughput",International Journal of Advanced Engineering Sciences and Technologies,Vol No. 6,Issue No.1,pp 22-26,2011.

[16] Iyer.N.C, Anandmohan.P.V, Poornaiah.D.V and Kulkarni.V.D, "High throughput, low cost, fully pipelined architecture for AES crypto chip", India Conference, Annual IEEE, pp 1-6,Sept.2006.

[17] T.Rahman,S. Pan and Q.Zhang, "Design of a high throughput 128-bit AES",Proc. Of the International Multiconference of Engineers and Computer Scientists,Vol.2,Hong Kong,March 17-19,2010.