Area Efficient Implementation Of Advanced Encryption System Using Vedic Mathematics

Haby K. Bose
Electronics And Communication
Rajagiri School of Engineering And Technology
Cochin, India
Email: haby.bose03@gmail.com

Dhanesh M.S
Electronics And Communication
Rajagiri School of Engineering And Technology
Cochin, India
Email:dhaneshmuralee@gmail.com

Abstract—**This work describes about the designing of Advanced Encryption System suitable for areas requiring maximal area minimization such as that for mobile phones. As the demand for secure transactions in banking and such related areas is increasing, encryption and decryption using cryptography plays a very important role. Nowadays, as majority of secure transactions occurs on smart phones and other handheld devices, an algorithm that consumes less area and that without compromising with overall performance becomes a necessity.In order to meet this requirement, several algorithms have been designed and implemented in the past, but each of these algorithms possess their own shortcomings with respect to an ASIC or an FPGA implementation. The design is done using Verilog hardware description language which provides an immediate hardware implementation possibility. The hardware implementation of the system is faster when compared to the conventional designs. We utilize the techniques involved in Vedic mathematics to realize the same. Comparisons are carried out with the conventional designs to state the advantages of the proposed design.**

Keywords—AES, Urdhwa Tiryakbhyam Sutra, Galois field multiplication.

I. INTRODUCTION

The encryption of data that is to be transmitted is always of major concern in wireless communication systems. Cryptographic algorithms have been proposed to encrypt and decrypt data to ensure security. It is useful to transmit and store data through insecure networks. In 2001, National Institute of Standard and Technology (NIST) replaced previous encryption standards like DES and Triple DES with Advanced Encryption Standard because of its efficiency, implementation and flexibility. The Advanced Encryption Standard is a subset of much larger encryption algorithm known as Rijndael.

The cryptographic algorithms involves encrypting the data to be transmitted or shared by means of unique keys, for encryption and decryption, which are known only to the authorized parties, thereby ensuring data security. This serves as a great boon to common man as well as for military applications.

Several cryptographic algorithms have been discovered and researched upon in the recent times, giving importance to the problem of vulnerability of the algorithms especially in applications which demand high security i.e. for smart cards, ATMs, WWW servers etc. [3]. Among these, the Advanced Encryption Standard (AES) algorithm is one of the highly preferred algorithms as it has higher immunity towards attacks [4].

However, when considering the hardware implementation of the design, the AES is losing, since it involves several complex operations implemented in the Galois Field (GF-2^8) [5]. Also, these complex operations are iterative in nature which in turn disturbs the speed of the encryption system and therefore increases the vulnerability.

In this paper, an area efficient architecture for performing the various operations involved in the Advanced Encryption Standard (AES) method of cryptography is introduced. Here, we make use of techniques used in ancient Vedic mathematics [7]. Vedic Mathematics is an archaic style of mathematics which subsisted in India in 1500 B.C., and was later on brought to limelight by a famous scholar Sri Bharthi Krishna Tirthaji between 1911 and 1918 [8]. He systemized it into 16 simple sutras [9], which are used by most of the researchers and mathematicians due to its ease of use. Out of the 16 formulae available in Vedic Mathematics, the Urdhwa Tiryakbhyam Sutra was utilized in order to address the flaws observed in the conventional mix columns architecture utilized in AES.

II. OVERVIEW OF AES ALGORITHM

The AES algorithm is a symmetric cipher. In symmetric ciphers, a single secret key is used for both the encryption and decryption, whereas in asymmetric ciphers, there are two sets of keys known as private and public keys. The plaintext is encrypted using the public key and can only be decrypted using the private key. In addition, the AES algorithm is a block cipher as it operates on fixed-length groups of bits (blocks), whereas in stream ciphers, the plain text bits are encrypted one at a time, and the set of transformations applied to successive bits may vary during the encryption process.

The basic block representation of Advanced Encryption System is shown in fig. 1. The input to the system is the 128 bit data to be encrypted and the secret Cipher Key for the AES algorithm. After the encryption operations we obtain the 128 bit encrypted data output. The decryption algorithm involves the inverse operations on the encrypted data to obtain the original 128 bit data. It is faster in both software and hardware implementations. By using hardware, a higher data rate for fast applications such as routers can be achieved compared to software implementation. The hardware implementation is also physically secure since tempering by an attacker is difficult. The efficiency of AES hardware implementation in terms of size, speed, security and power consumption depends largely on the AES architecture.

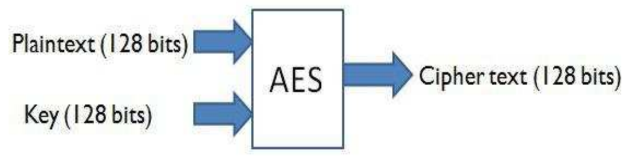


Fig.1Basic block representation of AES

In AES, fixed-length group of data bits are operated on for encryption and thus it is called as block cipher. AES has a block size of 128 bits and a key size of 128, 192 or 256 bits which results in 10, 12 or 14 rounds of operation respectively. After these rounds of operation, each bit of cipher text depends on each bit of plain text. Unlike the Data Encryption Standard, the decryption algorithm differs from encryption in AES.The 128 bit data to be encrypted is considered as a 4x4 matrix called the state array,with each element being one byte of data. The various operations are carried out on eachelement or an entire row/column of this state array to obtain the encrypted output. The state array is shown in fig. 2.

$$\begin{bmatrix} \text{byte}_0 & \text{byte}_4 & \text{byte}_8 & \text{byte}_{12} \\ \text{byte}_1 & \text{byte}_5 & \text{byte}_9 & \text{byte}_{13} \\ \text{byte}_2 & \text{byte}_6 & \text{byte}_{10} & \text{byte}_{14} \\ \text{byte}_3 & \text{byte}_7 & \text{byte}_{11} & \text{byte}_{15} \end{bmatrix}$$

Fig.2Representation of data in State Array

A. Algorithm Description

The state array is modified at each stage of encryption or decryption. For key size of 128 bit, 10 rounds of operations are involved. The output state array produced by the last round is rearranged into a 128-bit output block. The complete operations involvedin the 10 round encryption as well as decryption are shown in figure 3. The left side of the key schedule represents the encryption and the other one is the decryption algorithm. For both encryption as well as decryption, the 10th round is different from the rest. It does not have the mix column step.

B. Sub bytes

This step consists of using a 16 x 16 lookup table to find a replacement byte for a given byte in the input state array. The entries in the lookup table are created by usingthe notions of multiplicative inverses in the Galois Field 2^8 and bit scrambling carried outto destroy the bit-level correlations inside each byte. The Sub byte operation is shown in fig 4.

C. Shift rows

The ShiftRows operation is a linear diffusion process, operating on individual rows.The goal of this operation is to scramble the byte order inside each 128-bit block. It cyclically shifts the last three rows of the state by different offsets.

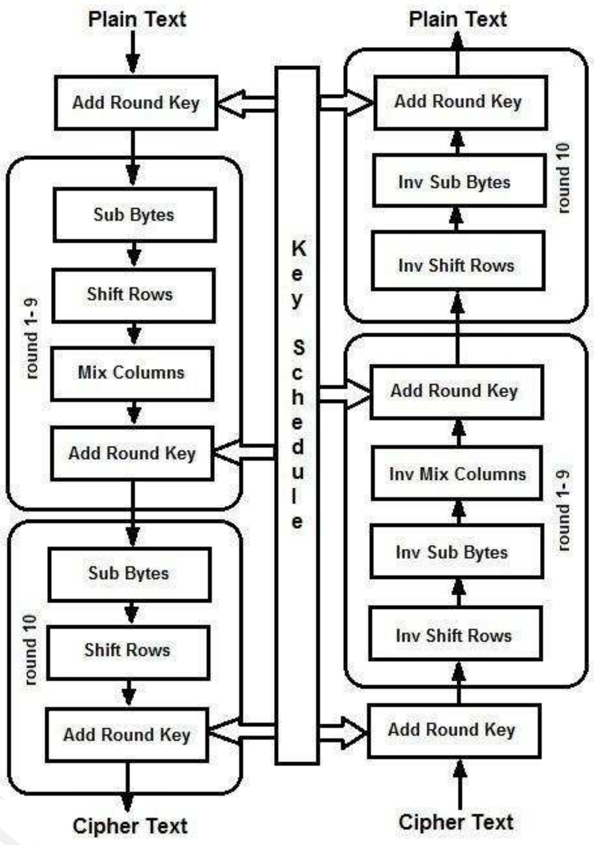


Fig.3.AES Algorithm

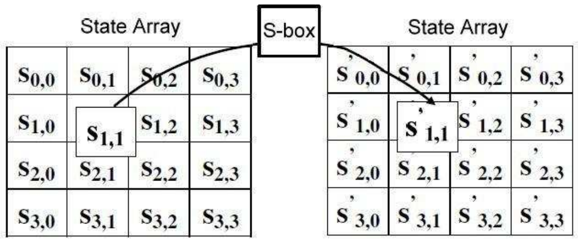


Fig.4 Transformation in Sub bytes

The first row is left unchanged in this transformation. Each of the byte in the second row is shifted one position to the left. The third and fourth rows are shifted left by two and three positions, respectively. Inverse shift rows is the inverse process of Shift rows transformation in which the bytes in the last three rows of the State are cyclically shifted over different numbers of steps to right. The Shift row operation is shown in fig 5.

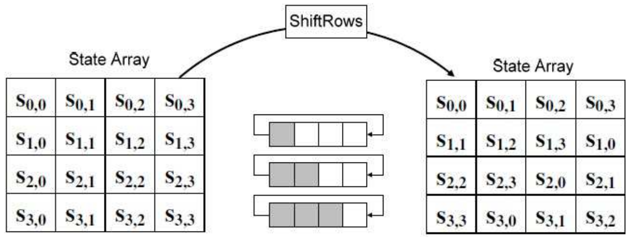


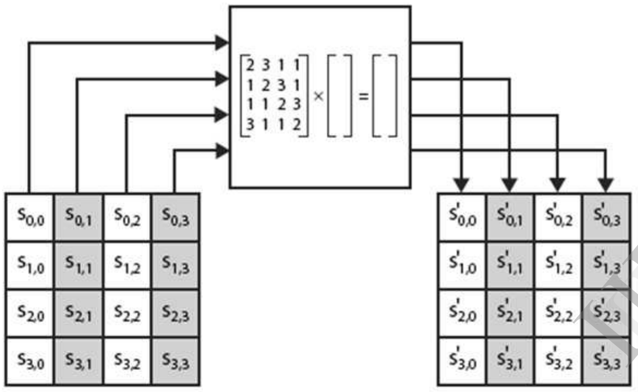
Fig.5 Shift Row operation

D. Mix columns

This transformation operates on the columns of the State, treating each columns asa four term polynomial the finite field GF(2^8). Each column is multiplied modulo x^4+1 with a fixed four-term polynomial.

$$a(x) = 03x^3 + 01x^2 + 01x + 02 \text{ over the } GF(2^8).$$

This step replaces each byte of a column by a function of all the bytes in the same column. The Mix Columns transformation can be expressed as a matrix multiplication as shown in figure 6. Here the Column vector is multiplied with a fixed matrix where the bytes are treated as a polynomials rather than numbers. In encryption process, the columns are multiplied with a matrix of the same size, while in decryption process, the columns are multiplied with another matrix. More precisely, each byte in a column is replaced by two times that byte, plus three times the next byte, plus the byte that comes next, plus the byte that follows. The shift rows step along with the mix column step causes each bit of the cipher text to depend on every bit of the plain text after 10 rounds of processing.



$\begin{bmatrix}2&3&1&1\\1&2&3&1\\1&1&2&3\\3&1&1&2\end{bmatrix} \times [\ ] = [\ ]$
$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$
$s_{1,0}$ $s_{1,1}$ $s_{1,2}$ $s_{1,3}$
$s_{2,0}$ $s_{2,1}$ $s_{2,2}$ $s_{2,3}$
$s_{3,0}$ $s_{3,1}$ $s_{3,2}$ $s_{3,3}$
$s'_{0,0}$ $s'_{0,1}$ $s'_{0,2}$ $s'_{0,3}$
$s'_{1,0}$ $s'_{1,1}$ $s'_{1,2}$ $s'_{1,3}$
$s'_{2,0}$ $s'_{2,1}$ $s'_{2,2}$ $s'_{2,3}$
$s'_{3,0}$ $s'_{3,1}$ $s'_{3,2}$ $s'_{3,3}$

Fig.6. Mixed Column operation.

E. Add Round Key

In this, each byte of the state array is combined with the round key. Each round keyis derived from the cipher key using a key expansion schedule. The round key is added to the state before starting the loop. In this step, each byte of the key state is combined with a byte of the round sub key using the XOR operation.

F. Key Expansion

The algorithm expands the four 4 byte words in each column of the key matrix intoa 44-word key schedule that can be labeled w0, w1, w2, w3, .., w43. The key expansion takes place on a four-word to four-word basis, in the sense that each grouping of four words decides what the next grouping of four words will be.

III. Mix column using Vedic mahematics

One of the crucial mathematical operation performed during the mix column step in AES, is the Galois field multiplication. Multiplication, being a tedious and a power hungry operation, causes the computation of mix columns and its inverse to be an even more difficult task. This is due to the fact that, it involves matrix multiplication. Therefore, there arises a necessity to ease the entire process of mix columns. In order to achievethe same, the Urdhwa Tiryakbhyam Sutra of Vedic Mathematics, is utilized in our proposed architecture for mix columns and its inverse due to its excellence in terms of speed and area.

The Urdhwa Tiryakbhyam Sutra is one of the significant sutras in ancient Vedic Mathematics. By its definition, Urdhwa Tiryakbhyam means “vertically crosswise". This implies that multiplication occurs between extreme bits of the multiplier and multiplicand.The major advantage of this algorithm is the availability of the product of two numbers in a single step. Also, since multiplication of two single bits reduces to a single AND operation, for a VLSI implementation this approach proves to be both faster and area efficient.

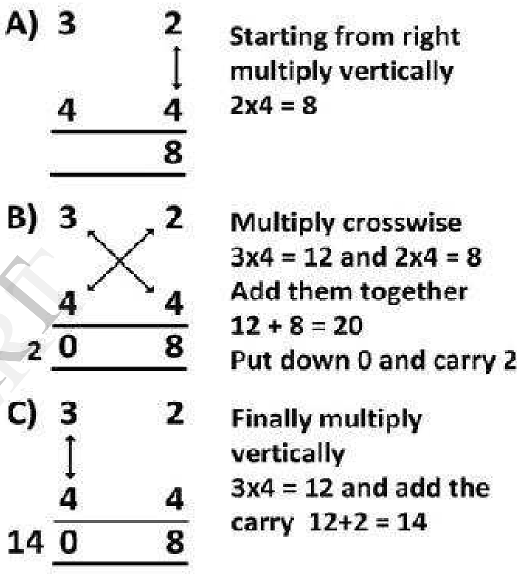


Fig.7. Vedic Multiplication

A. Vedic Multiplication for Binary Numbers

Let us assume M and N are the two 8 bit numbers to be multiplied. Let us assume P is the product achieved upon multiplication of M & N. The subscripts to both the letters M & N indicate the bit positions in both the numbers. The subscript ‘0’ indicates the Least Significant Bit (LSB) and the highest subscript indicates the Most Significant Bit (MSB). The Urdhwa Tiryakbhyam Sutra essentially comprises of logically cross-ANDing the bits of the multiplier and the multiplicand. These cross-ANDed results are termed as partial products. These partial products are immediately added so as to calculate each bit of the product. It must also be noted, that the addition of these partial products leads to the generation of carry’s, denoted by Ci where i=0, 1, 2 to 30. The equations to obtain each bit of the product have been represented by 1 to 16.

$$P0= M0 * N0 \qquad (1)$$

$$C1P1= (M1 * N0) + (M0 * N1) \qquad (2)$$

C3C2P2= (M2 ∗ N0) + ((M0 ∗ N2)+ (M1∗ N1) +C1 (3)

*C5C4P3 = (M3 * N0) + (M2 * N1) + (M1* N2) + (M0 * N3) + C2* (4)

*C7C6P4 = (M4 * N0)+(M3* N1) +(M2 * N2) + (M1 * N3) +(M0 * N4) +C3 + C4* (5)

*C10C9C8P5 = (M5 * N0) + (M4* N1) +(M3 * N2) + (M2 * N3) +(M1 * N4) + (M0 * N5) +C5 + C6* (6)

*C13C12C11P6 = (M6 * N0) + (M5* N1) +(M4 * N2) + (M3 * N3) +(M2 * N4) +(M1 * N5) +(M0 * N6) +C7 + C8* (7)

*C16C15C14P7 = (M7 * N0) + (M6* N1) +(M5 * N2) + (M4 * N3) +(M2 * N5) +(M1 * N6) +(M0 * N7) +C9 + C11* (8)

*C19C18C17P8 = (M7 * N1) + (M6 * N2) +(M5 * N3) + (M4 * N4) +(M3 * N5) +(M2* N6) +M1* N7) + C10 +C12 + C14* (9)

*C22C21C20P9 = (M7 * N2) + (M6 * N3) +(M5 * N4) + (M4 * N5) +(M3 * N6) + (M2* N7) + C13 +C15 +C17* (10)

*C25C24C23P10 = (M7 * N3) +(M6* N4) +(M5 * N5) + (M4 * N6) +(M3 * N7) +C16 + C18 + C20* (11)

*C27C26P11 = (M7 * N4) + (M6 * N5) +(M5 * N6) + (M4 * N7) + C19 +C21 + C23* (12)

*C29C28P12 = (M7 * N5) + (M5 * N6) +(M5 * N7) + C22 + C24 +C26* (13)

*C30P13= (M7 * N6) + (M6 * N7)+C25 +C27 + C28* (14)

*P14 = (M7 * N7) + C29 + C30* (15)

*P15= (M7 * B7)* (16)

B. *Simplification of Vedic Method for Galois field multiplication*

The above mentioned Urdhwa Tiryakbhyam sutra for multiplication can be used in the Mix columns operation of AES. As Galois Field multiplication needs to be carried out instead of the regular multiplication, a small variation for the proposed architecture has to be done from this Vedic architecture. In our implementation, instead of adding the partial products that is computed in the intermediate stages, logically XOR of the same is carried out. The product is limited to GF(2^8) upon performing multiplication using this algorithm.

IV. RESULTS AND SIMULATION

The algorithm for the Advanced Encryption Standard using Vedic Mathematics technique was designed and simulated in Verilog HDL using Xilinx ISE.The design of the conventional implementation of the AES was also carried out in order to compare with the proposed design. The designing was carried out so as to be implemented and synthesized for a Spartan 3e series XC3s1600e Xilinx FPGA. The table below shows the results of the simulation proposed design, after comparison with the conventional counterpart.

TABLE I. COMPARISON OF ALGORITHMS

Algorithm	**Area Occupancy(%)**	**No. of gates**	**Timing(ns)**
Conventional AES	7.35	1210	11.816
Proposed AES using Vedic Technique	5.11	731	12.33

We can find that the proposed design provides savings in overall area required for implementation when compared with the conventional design, with only a small increase in the timing requirement.

CONCLUSION

In this paper, an area efficient design of 128 bit advanced encryption standard that is suitable for carrying out cryptographic applications is done. The architecture of design performs well when compared withthe conventional designs. The design provided good savings in the overall area with only neglectable increment in the timing requirementthat proves its applicability in mobile devices. The optimizations can be extended towards the design of Sub byte operation in the future.

References

[1] Huang, Xu, Shirantha Wijesekera, and Dharmendra Sharma. "Quantum cryptography for wireless network communications." Wireless Pervasive Computing, 2009. ISWPC 2009. 4th International Symposium on. IEEE, 2009.

[2] William Stallings, "Cryptography and Network Security: Principles and Practice",3rd ed. Pearson Education, 2002.

[3] Knudsen, Lars R., and Matt JB Robshaw, “The block cipher companion”,Springer, 2011.

[4] Rady, A., E. El Sehely, and A. M. El Hennawy. "Design and implementation of area optimized AES algorithm on reconfigurable FPGA," Microelectronics, . ICM 2007. Internatonal Conference on, pp. 35-38. IEEE, 2007.

[5] Rachh, Rashmi Ramesh, PV Ananda Mohan, and B. S. Anami. "Efficient Implementations for AES Encryption and Decryption." Circuits, Systems,and Signal Processing 31, no. 5 (2012): 1765-1785..

[6] Zhang, Xinmiao, and Keshab K. Parhi. "Implementation approaches for the advanced encryption standard algorithm." Circuits and Systems Magazine, IEEE 2.4 (2002): 24-46

[7] Huddar, S.R.; Rupanagudi, S.R.; Kalpana, M.; Mohan, S., "Novel highspeed vedic mathematics multiplier using compressors," in Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013 International Multi-Conference on , vol., no., pp.465,469, 22-23 March 2013

[8] Jagadguru Swami Sri Bharati Krisna Tirthaji Maharaja, “Vedic Mathematics: Sixteen Simple Mathematical Formulae from the Veda,” pp. 5-45, Motilal Banarasidas Publishers, Delhi, 2009.

[9] Huddar, S.R.; Rupanagudi, S. R.; Janardhan, V.; Mohan, S.; Sandya, S.,"Area and Speed Efficient Arithmetic Logic Unit Design Using Ancient Vedic Mathematics on FPGA,"in Advances in Computing, Communication, and Control, pp. 475-483, Springer, Berlin Heidelberg, 2013.

[10] FIPS, PUB. "197." Advanced Encryption Standard (AES) 26 (2001)