DESIGN AND IMPLEMENTATION OF EFFICIENT ADVANCED ENCRYPTION STANDARD COMPOSITE S-BOX WITH CM-MODE

S. Gnana Soundari, B. Senthil Kumar

ECE Sri Venkateswara College of Engineering & Technology, Chittoor, India
soundarivs@gmail.com; senb2002@gmail.com

ABSTRACT

The Advanced Encryption Standard (AES) is one of the famous algorithms for the cryptographic security algorithm and it is specifically used for the data protection and the transmission. The AES novel Mix-Column operation implementation is proposed in this research paper. The Mix-Column is model is improved for the AES decryption through the Very Large Scale Integration (VLSI) System design in this research paper. In AES Mix-Column, large number of logic gates used to perform the multiplication of input stage bytes (output of shift row) and fixed defined state bytes. In order to decrease this problem, the redundant function of Mix-Column is eliminated and re-designed in this paper. Proposed model of Mix-Column minimizes 25% of logic gates compared with previous work. Further, the proposed Mix-Column of AES decryption achieves by improving the performance of area, delay and power consumption. The transformations optimized and the speed is increased.

Index Terms—Advanced Encryption Standard (AES), Rijindael Algorithm, Enhanced Mix-Column, S-box, Composite Field Arithmetic (CFA), Very Large Scale Integration (VLSI).

I. INTRODUCTION

In the cryptography algorithm, one of the significant algorithms is Advanced Encryption Standard. This algorithm is specifically designed for the electronic data encryption process and the US NIST (National Institute of Standards and Technology) is established this algorithm in 2001. In the Rijndael Cipher, this method is established and this process is achieved by using the two Belgian cryptographers, Daemen and Vincent [1] submitted proposal to NIST while the Advanced Encryption Standard for the selection process. Rijndael is a family of ciphers with different block sizes and key. AES is having 128 bits of block size and three different key lengths for cipher such as 128 bits, 192 bits and 256 bits. Finally AES is accepted by US government and it is used worldwide now. Before AES design, Data Encryption Standard is used and it has asymmetric key. There are two various types of keys are required for the data encryption process and also it is required for the decryption process.

But AES is symmetric key algorithm; both the encryption and decryption the common key is cipher key. The paper describes the AES-128 Rijendael algorithm for symmetric key encryption and it's selected by the sensor networks. In the 8-bit Microcontroller, the encryption and decryption performance is explained [8, 9]. After that, in the sensor networks the perfor-mance is analyzed the communication efficiency over the total delay per hop.

II. RIJINDAEL'S ALGORITHM

One of the encryption standards is the Advanced Encryption Standard (AES) [3] as a symmetric block cipher. This encryption standard is established by National Institute of Standards and Technology (NIS T) as U.S. FIPS PUB 197 (FIPS 197) on November 26, 2001.

The Rijndael algorithm is selected by the NIST after 5 years as the AES. The AES process has the 128-bit blocks of data. This algorithm is mainly used for the encoding and decoding process of the blocks by utilizing the secret keys. The secret key size can either be 128 bit, 192 bit, or 256 bit. The actual key size is relies on the preferred on the security level. Frequently, the different versions are denoted as AES-128, AES-192 or AES-256. There are three types are available in the cipher Rijndael [4] namely initial Round Key addition, Nr-1 Rounds, a final round. By using these types the pseudo C code of Rijndael algorithm [7] is given.

```
Rijendael (State, Expanded Key)
{Key Expansion (Cipher Key, Expanded Key);
        Add Round Key (State, Expanded Key);
For (i=1; i<Nr; i++) Round (State, Expanded Key +
Nb * i); Final Round (State, Expanded Key+Nb*Nr);}
```

By using the expanded key the Rijndael can be specified and the expansion of the key is also completed before the execution. The expanded key can be determined from the cipher key and it does not directly state. On the other hand, there is no restriction to select the cipher key. The pseudo C code of Rijndael Expanded Key algorithm is written as follows.

```
Rijendael (State, Expanded Key)
{ Add Round Key (State, Expanded Key);
For(i=1;i<Nr;i++) Round(State, Expanded Key+Nb*
i); Final Round(State, Expanded Key+Nb*Nr); }
```

Further, S-box, Shift Rows, Mix Columns and Add Round Key methods are done in encryption process and reverse process is followed by decryption process. In those different levels of modes are available for encrypt and decrypt data with high level of performance.

III. COUNTER MODE

AES Counter Mode (CM) mode of operation doesn't utilize the AES cipher block directly to encode the data like ECB or CBC do; in a contrary, it encrypts an arbitrary value named as counter and after that the XORs result with the real data to produce the ciphered text. Usually, the counter value is incremented by one for the entire efficient block processing.

The information is divided into 128 bit vectors. The entire vector is XORed with the result of the encrypting the counter value is same as to that block by utilizing an AES at 1 and added by one up to 4 and the 512 bits are crypts in parallel.

The receiver decrypts the data by utilizing the same circuit and it should know the beginning value of the counter in advance.

IV. Implementation of Enhanced Sub-Byte Transformation

The significant contributions can be summarized as follows. Figure.1 represents the execution of Sub Bytes. This paper eliminates the utilization of LUTs and proposes the utilization of composite field data path for the Sub Bytes and InvSubBytes transformation.

Composite field arithmetic is working to design effective data paths. On the other hand, the inversion is composed in the Sub Bytes and InvSubBytes. In-stead, the presented architecture in this paper reduce the multiplicative inverse (MI) architecture. Further, this structure is incorporated into multiplicative inverse block of composite SubBytes transformation.

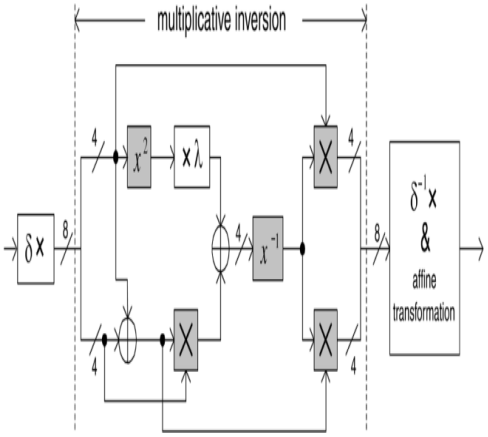

Fig.-1: Implementation of the Sub Bytes Transformation

V. Composite Field Arithmetic

Composite field often utilized in the execution of Galois field arithmetic. In the AES algorithm, the non-LUT based execution can use the advantages of sub pipelining. On the other hand, these methodologies have the complexities of the hardware. Even though, two Galois fields of the similar sequence are isomorphic, the complexities of the field operations is depend on the presentation of the field components. The cost of the transformation is based on the composite field choice. The composite field arithmetic operations can be managed to reduce the difficulties of the hardware. The composite S-Box into the AES-CM algorithm is executed to acquiring area and delay.

VI. Existing Mix-Column Transformation for AES

The State column-by-column is worked based on the Mix-Column transformation, and the entire column as in the form of four term polynomials. These columns are received as polynomials through the GF (28) and multiplied by modulo x4+1 with a fixed polynomial x (n) is expressed by,

$$x(n) = \{03\}\, x^3 + \{01\}\, x^2 + \{01\}\, x + \{02\}$$

The matrix multiplication can be expressed as s'(x) = x(n)*s(x), where s(x) and it is represented the state byte from Shift Row method. These can be given as,

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < \boldsymbol{Nb}.$$

Similarly, inverse Mix-Column can be computed by utilizing x-1(n). The columns are received as polynomials through GF(2^8) and multiplied modulo x4+ 1 with a fixed polynomial x^{-1}(n). Likewise, the inverse Mix-Columns can be determined by utilizing x^{-1}(n). The columns are received as polynomials through GF (2^8) and multiplied modulox4+ 1 with a fixed polynomial x^{-1}(n).

$$x^{-1}(n) = \{0b\}\, x^3 + \{0d\}\, x^2 + \{09\}\, x + \{0e\}$$

Therefore s'(x)=x^{-1}(n)⊕ s(x). The matrix for InvMix-Column can be denoted as,

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < \boldsymbol{Nb}.$$

The multiplication of input state bytes with fixed pre-defined polynomials can be processed by Xtime (Both input and output word length of multiplication is same) multiplication. In every multiplication step, set of inputs are logically surrounded by EX-OR gate.

The process of Xtime multiplication for the Inverse Mix-Column transformation is explained in fig.2. Matrix for InvMix-Column demands more number of logic gates to perform multiplication than Mix-Column because of long word length. Typically, 24 EX-OR gates are utilized to perform the Inv Mix-Column operation in AES.

Because of using more number of logic gates, area for existing InvMix-Column use large area and delay. To minimize this issue, circuits for InvMix-Column is fulfilled in this paper. In next section Enhanced InvMix-Column is described in detailed manner.

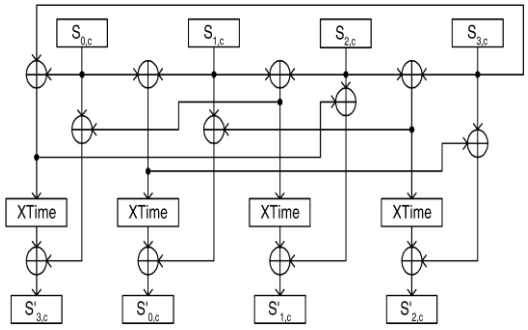

Fig. 2. Xtime Multiplication circuit for InvMix-Column Transformation

VII. Enhanced Mix-Column Transformation for AES

When compared to Mix-Column transformation, InvMix-Colum transformation has multiplication of long word length. Therefore, Matrix multiplication of

InvMix-Column can be realized and re-designed. Design of Optimized InvMix-Column is represented in below. In InvMix-Column, {09, 0b, 0d, 0e} is multiplied with input bytes. Let input bytes are represented as b0 to b7 and output of InvMix-Column are represented as t0 to t7. The multiplication can be described as follows:

Multiplication of {09} with state-byte,

$t_7 = 0,\ t_6 = b_7, t_5 = b_6 \oplus b_7, t_4 = b_5 \oplus b_6,$
$t_3 = b_5 \oplus b_7,\ t_2 = t_5, t_1 = t_4, t_0 = b_5$

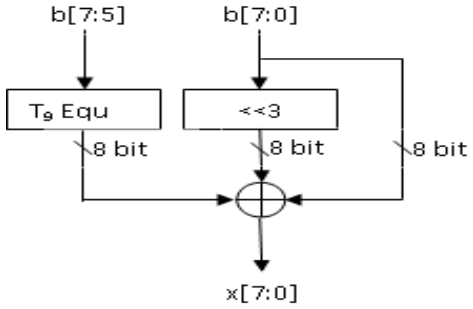

Fig. 3 Advanced Xtime for 09

Multiplication of {0b} with state-byte,

$t_7 = 0,\ t_6 = b_7,\ t_5 = b_6 \oplus b_7, t_4 = b_5 \oplus t_5,$
$t_3 = b_5, t_2 = t_5, t_1 = t_4, t_0 = b_5 \oplus b_7$

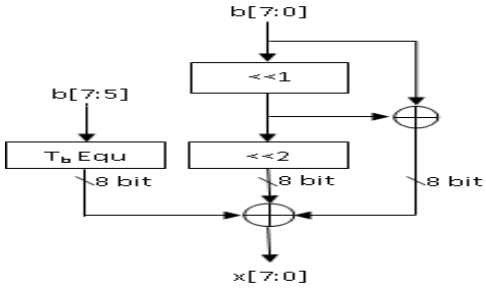

Fig. 4 Advanced Xtime for 0b

Multiplication of {0d} with state-byte,

$t_7 = 0,\ t_6 = b_7,\ t_5 = b_6, t_4 = b_5 \oplus b_7,$
$t_3 = b_5 \oplus t_4, t_2 = b_6, t_1 = t_4, t_0 = b_5 \oplus b_6$

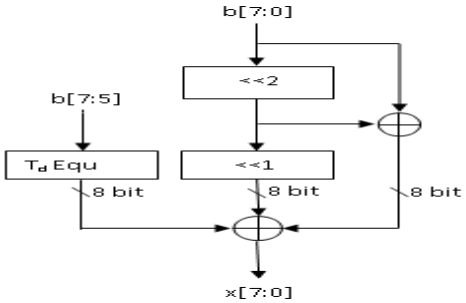

Fig. 5 Advanced Xtime for 0d

Multiplication of {0e} with state-byte,

$t_7 = 0\ ,\ t_6 = b_7, t_5 = b_6, t_4 = b_5, t_3 = b_5 \oplus b_6,$
$t_2 = b_6, t_1 = b_5, t_0 = t_3 \oplus b_7$

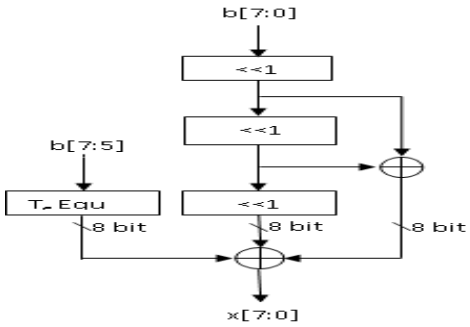

Fig. 6 Advanced Xtime for 0e

The circuit diagram for optimized Mix- Column for above equation representation is illustrated in fig. 3, fig. 4, fig. 5and fig. 6 correspondingly, in this case, the less number of EX-OR gates are used to execute the matrix multiplications. In this optimized InvMix-Column design, 13 number of EX-OR gates are used instead of 24 number of EX-OR gates. Hence, hardware complexity InvMix-Column can be reduced significantly than existing one. Moreover, the InvMix-Column is optimized and it is incorporated into AES decryption process for to enhance the performance.

VIII. Hardware Implementation

The structural design for the entire non-trivial transformation in the AES algorithm is explained [2, 5, 6]. The process of the entire transformation is optimized to minimize the area and raise the speed. Meanwhile, efficiency key expansion architecture appropriate for round units is presented. According to the analysis on the gate counts in the critical path of the round units and the key expansion of the AES algorithm are introduced.

IX. Implementation Results

The The Verilog HDL implementation has been enhanced and the Xilinx FPGA devices are synthesized.

In FPGA, the synthesis process is completed with the Xilinx software tool 10.1. the simulation process is completed with the Model Sim XE 6.3c. The Advanced Encryption Standard (AES) algorithm is provided by using the Verilog DL language. Both the simulation and synthesis process is tested the program.

Simulation of the Model Sim XE 6.3c issued for investigating and for synthesizing the Xilinx ISE tool to be utilized. It shows the simulation result of the encryption module and fig 7 and 8 shows the encryption and decryption results respectively.

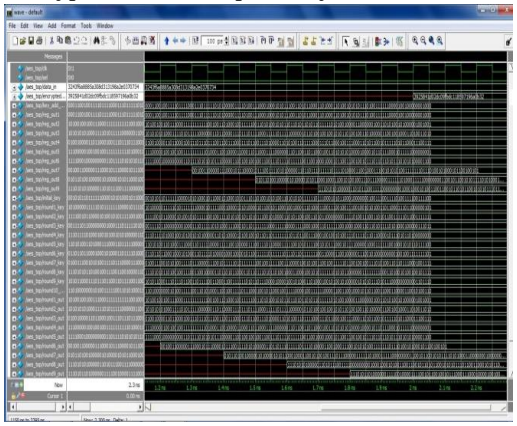

Fig. 7 AES Encryption Result

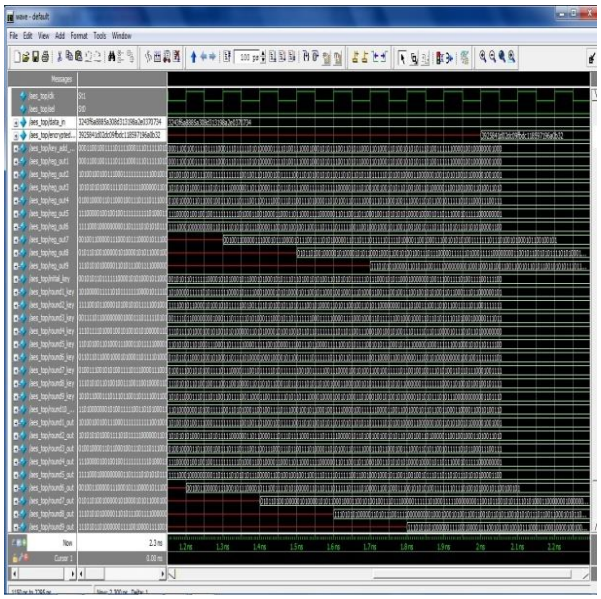

Table.1 Comparison of Existing and Proposed AES Decryption

Types	Area	Delay	Power
Existing AES using xtime multiplication based InvMix-Column	10750	8.686	7.785
Proposed AES using Enhanced InvMix-Column	10471	3.793	6.542

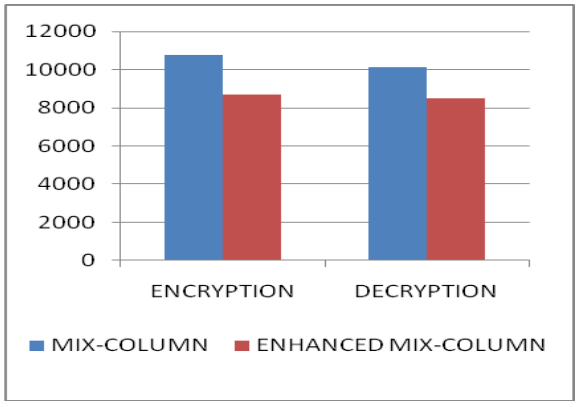

Fig. 9. Graphical Representation of Simulated Result for Area

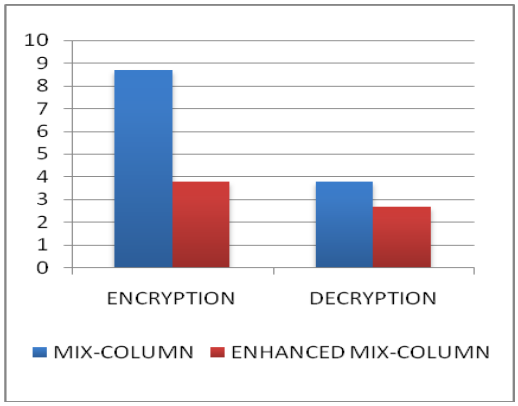

Fig .10. Graphical Representation of Simulated Result for Delay

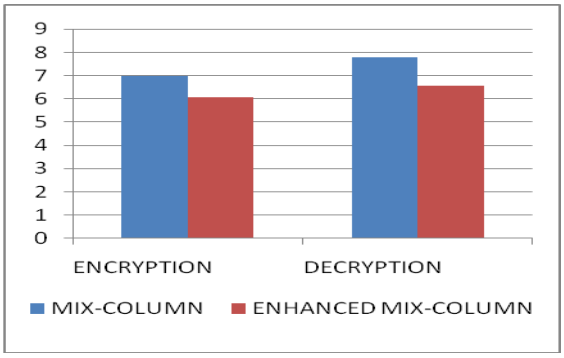

Fig. 11. Graphical Representation of Simulated Result for Power Consumption

X. Conclusion

The iterative form of private key symmetric block cipher is one of the Advanced Encryption Standard Algorithm. The InvMix-Column of the AES decryption is improved in this paper over the Very Large Scale Integration (VLSI) System design environment. The structure of Xtime multiplication can be optimized by reducing the logic gates. In Enhanced InvMix-Column design 9 gates are reduced to perform the Xtime multiplication. Further Enhanced InvMix-Column Transformation technique is incorporated into AES decryption for improve the performance of decryption. Proposed AES decryption gives 12.69% reduction in area, 22.9% reduction in delay and 11.90% reduction in power consumption than existing AES decryption. The AES decryption structural design is proposed and in future it is very helpful in Satellite, Space, and terrestrial and data communication application with respect to give the security and enhances the performance of hardware usage.

References

[1] DAEMEN, J., RIJMEN, V., AES Proposal: Rijndael, The Rijndael Block Cipher. AES Proposal pp.1–45(1999)

[2] High Speed Low Cost Implementation of Advanced Encryption Standard on FPGA- International Journal of Electronics & Telecommunication and Instrumentation Engineering (IJETIE), March (2010).

[3] Scalable 128-bit AES-CM Crypto-Core Reconfigurable Implementation for Secure Communications, IEEE conference (2010).

[4] Implementation and Performance Analysis ofAES-128 CBC algorithm in WSNs- Hyeopgeon Lee, Kyounghwa Lee, Yongtae Shin Department of Computing, Soongsil University, Korea.

[5] High-Speed VLSI Architectures for the AES Algorithm - EEE transactions on very large scale integration (vlsi) systems 12(9): (2004).

[6] Hardware Evaluation of the AES Finalists, in Proceedings of the Third Advanced Encryption Standard (AES) Candidate Conference pp. 297–285 (2000).

[7] Experimental Testing of the Gigabit IPSec-Compliant Implementations of Rijndael and Triple DES Using SLAAC-1V FPGA Accelerator Board in Proceedings of the Information Security Conference pp. 220–234 (2001).

[8] Design of an Extremely High Performance Counter Mode AES Reconfigurable Processor, in Proceedings of the Second International Conference on Embedded Software and Systems (ICESS'05) Pp. 262–268 (2005).

[9] FPGA Implementation AES for CCM Mode Encryption Using Xilinx Spartan-II, ECE 679, Advanced Cryptography, Oregon State University (2003).