DESIGN OF ADVADVANCED ENCRYPTION STANDARD (AES) BASED RIJINDAEL ALGORITHM

J.Aarthi,
Electronics and Communication Engineering,
Jayaram College of Engineering and Technology, Trichy
Mail Id: aarthirachagan@yahoo.com

Abstract - Advanced standard encryption standard is mainly used for various security applications. Application specific integrated circuit (ASIC) with cryptographic algorithm is generate the efficient security levels for various applications. Rijindael algorithm works with encryptions and decryptions. Encryption is transform the transformations from plain text into cipher text and decryption are performs the reverse function of encryption. Mix column and inverse mix column is one of the complicate process to perform the encryptions and decryptions. In general, there are required number of the logical elements is comparatively high for the mix column and inverse mix column process. To provide the efficient logical elements for the mix column, the enhanced inverse mix column is proposed in this paper. These works improve the logical elements utilizations as well as the power optimizations. The enhanced mix column is produce the efficient Area, Delay and Power(ADP) products for the both encryption and decryptions.

Keywords - Advanced Standard Encryption (AES), Substitution Bytes(S-Box), Application-Specific-Integrated-Circuit (ASIC), Area, Delay, Power (ADP).

1. Introduction

To protect the classified information, the symmetric block cipher is utilized for advanced encryption standard. For rijndael algorithm, there three size of bits is used for operations such as 128 Bits, 192 bits and 256 bits. Daemon implement the rijndael algorithm with 32 bit of the data blocks to provide efficient executions of the algorithm. Multiplicative masking and the Boolean masking is the main representatives of the AES algorithm. Composite S-Box is used to reduce the hardware utilizations and power utilizations as well as it improve the mix column transformations. Enhance mix column with composite s-box is evaluated and analyzed in terms of VLSI factors such as Area, Delay and Power.

2. Related Works

Application specific integrated circuit(ASIC) with cryptographic algorithm is provide better security levels for various applications [1] described the AES Algorithm with efficient Mix column transformations.. Byte level representations are performed for standard operations. Galois field with polynomial is used to assume the bytes of the representations. AES standard

hardware implementation is high for the decryption process. To reduce the complexity of the both encryption and decryption is performed by the fixed coefficient technique. [2] presented the storage network for masked AES for reduce the utilizations of the storage network. For satellite communications and security applications, the AES can be used for security applications Pipelining architecture is improve the speed of the architecture and also for security applications. Application specific integrated circuit(ASIC) with cryptographic algorithm is provide better security levels for various applications. [4] described the incorporation of wave pipelined techniques into composite S-Box and AES architectures. For every round of the operations, the wave pipeline technique is used to provide the efficient S-Box. Clock gating technique is used for the implementations; these techniques minimize the delay as well as the area. The field information is mainly depending on the byte representations.[5] explained the Sub bytes transformations. For the sub bytes transformations, there is no need for the LUTs and field of the data path. Decryption is retrieving the given input information in final stage.

3. Rijndael Algorithm

Rijndael algorithm is used for the AES method for the security purpose to provide efficient secured information data from the decryption process. AES is depend on Rijndael algorithm choose for data encryption standard by National Institute of Standards and Technology (NIST) in 1997. It performs data blocks of fixed size by cipher keys of length 128, 196 and 256 bits . AES-128 bit cipher keys are mainly used for encryption and decryption. Encryption side of AES can operate the 4 discrete transformations in particular order like S-Box, Shift Rows, MixColumn and Add Round Key. There are Three numbers of rounds have to be processed in order of 10, 12 and 14 for AES-128, AES-196 and AES-256 bits respectively. S-Box is provided by taking multiplicative inverse of data input in the finite Galois Field GF (2^8) and it followed by an affine transformation. The irreducible polynomial of data input is represented as follows:

$$m(z) = z^8 + z^4 + z^3 + z + 1 \qquad \text{(I)}$$

.The matrix multiplication of MixColumn of AES is derived as below,

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 01 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \qquad \text{(II)}$$

Add Round Key function can be processed in a 10 rounds of transformations can be performed, since here AES-128 bit length is considered for both input and output. The standard flow chart of AES encryption and decryption is analyzed in fig. 1.

Matrix multiplication of Inv MixColumn of AES is derived as below,

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{(III)}$$

4. General Inverse Mix-Column Transformation

The Mix-Column transformation performs on the State column-by-column, each column as a four term polynomial considered as polynomials over GF (2^8) and multiplied by modulo z^4+1 with a fixed polynomial x (n) is given by,

$$z(n) = \{03\}z^3 + \{01\}z^2 + \{01\}z + 02 \quad \text{(I)}$$

This can be written as a, s' (z) = z (n)*s (z). These can be illustrated as,

$$\begin{bmatrix} S'_{0c} \\ S'_{1c} \\ S'_{2c} \\ S'_{3c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3.c} \end{bmatrix} \quad for\ 0 \le c < Nb \quad \text{(II)}$$

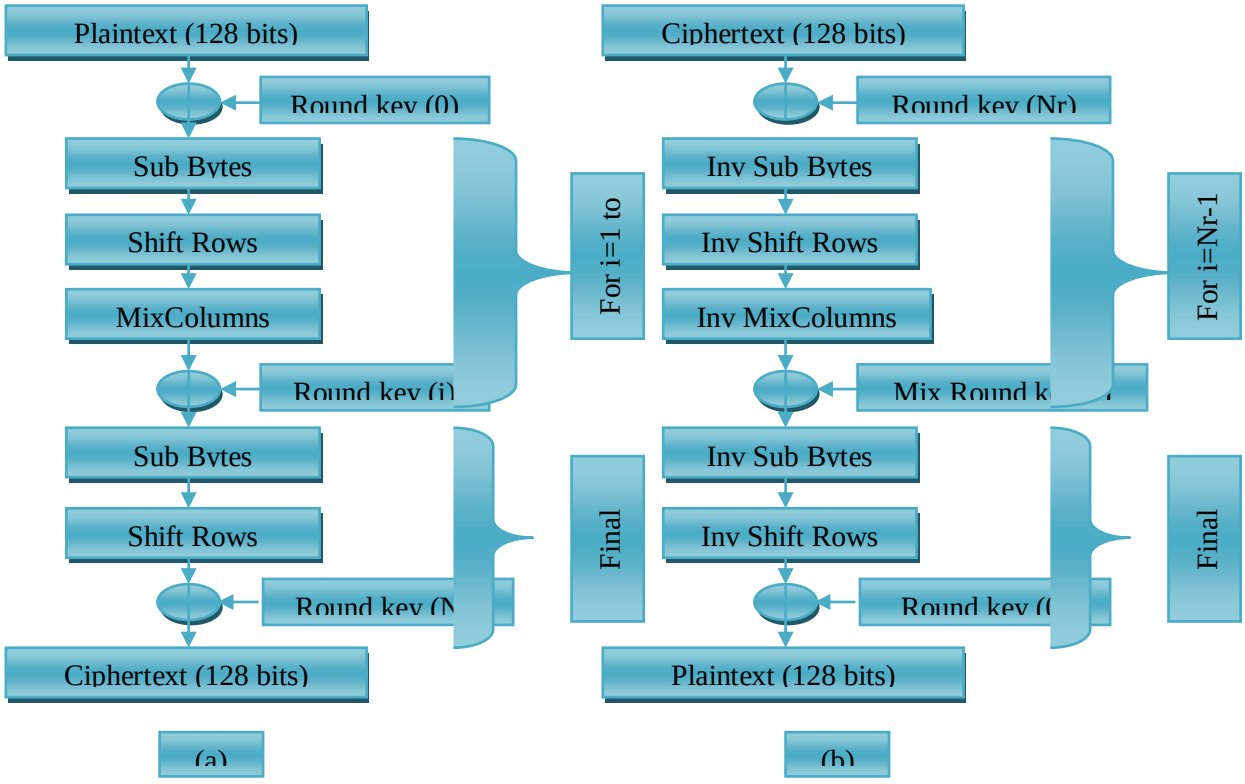

Fig. 1 AES ARCHITECTURE

Inverse mix column of the AES transformations illustrated as below,

$$z^{-1}(n) = \{0b\}z^3 + \{0d\}z^2 + \{09\}z + \{oe\} \qquad \text{(III)}$$

(i.e) $s'(z) = z^{-1}(n) \oplus s(z)$ The InvMix-Column matrix can be represented as below,

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \quad for\ 0 \le c < Nb \qquad \text{(IV)}$$

For Xtime multiplications, the standard polynomials input bytes are usedfor the multiplication process. These inputs are processed by using simple EX-OR gate operations that is by using usual logical exor gate, the bytes of input is performed. The required number of the logical gates are comparatively high for the inverse mix column process because of its word length. High utilizations of the logical gates is leads to the high area utilizations and also increase the power. To reduce the power as well as area, the inverse mix column circuits is proposed in this section.

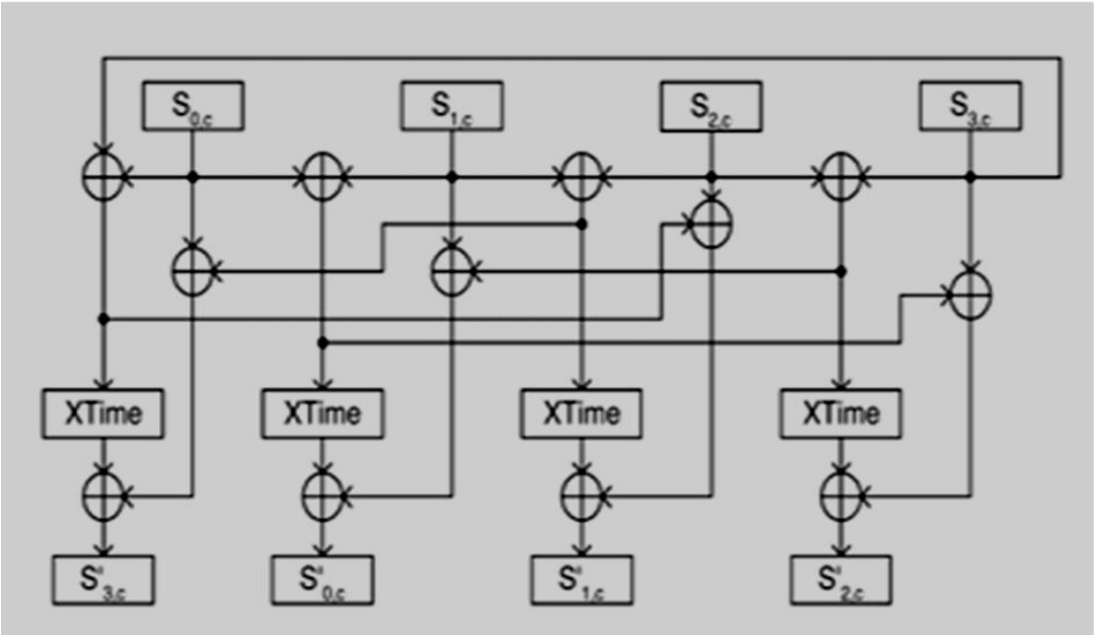

Fig. 2 Xtime Multiplcation

5. Proposed Enhanced Inv Mix-Column with Composite S-Box

To provide the efficient logical elements utilizations, the enhanced inverse mix column is proposed in this work. The proposed work is mainly focused to reduce the computational path of the equations. The proposed work is reducing the number of logical elements in the equations. Proposed equations are illustrated as below,

From equation 1,

$$s'_{0,c} = (\{0e\} * s_{0,c}) \oplus (\{0b\} * s_{1,c}) \oplus (\{0d\} * s_{2,c}) \oplus (\{09\} * s_{3,c}) \quad (5)$$

$$s'_{1,c} = (\{09\} * s_{0,c}) \oplus (\{0e\} * s_{1,c}) \oplus (\{0b\} * s_{2,c}) \oplus (\{0d\} * s_{3,c}) \quad (6)$$

$$s'_{2,c} = (\{0d\} * s_{0,c}) \oplus (\{09\} * s_{1,c}) \oplus (\{0e\} * s_{2,c}) \oplus (\{0b\} * s_{3,c}) \quad (7)$$

$$s'_{3,c} = (\{0b\} * s_{0,c}) \oplus (\{0d\} * s_{1,c}) \oplus (\{09\} * s_{2,c}) \oplus (\{0b\} * s_{3,c}) \quad (8)$$

Further, equation (5) to (8), can be simplified as

$$s'_{0,c} = [(\{09\} * s_{0,c}) \oplus (\{04\} * s_{0,c}) \oplus (\{02\} * s_{0,c}) \oplus s_{0,c}] \oplus [(\{09\} * s_{1,c}) \oplus (\{02\} * s_{1,c})] \oplus [(\{09\} * s_{2,c}) \oplus (\{04\} * s_{2,c})] \oplus [(\{09\} * s_{3,c})] \quad (9)$$

$$s'_{1,c} = [(\{09\} * s_{0,c})] \oplus [(\{09\} * s_{1,c}) \oplus (\{04\} * s_{1,c}) \oplus (\{02\} * s_{1,c}) \oplus s_{1,c}] \oplus [(\{09\} * s_{2,c}) \oplus (\{02\} * s_{2,c})] \oplus [(\{09\} * s_{3,c}) \oplus (\{04\} * s_{3,c})] \quad (10)$$

$$s'_{2,c} = [(\{09\} * s_{0,c}) \oplus (\{04\} * s_{0,c})] \oplus [(\{09\} * s_{1,c})] \oplus [(\{09\} * s_{2,c}) \oplus (\{04\} * s_{2,c}) \oplus (\{02\} * s_{2,c}) \oplus s_{2,c}] \oplus [(\{09\} * s_{3,c}) \oplus (\{02\} * s_{3,c})] \quad (11)$$

$$s'_{3,c} = [(\{09\} * s_{0,c}) \oplus (\{02\} * s_{0,c})] \oplus [(\{09\} * s_{1,c}) \oplus (\{04\} * s_{01c})] \oplus [(\{09\} * s_{2,c}) \oplus (\{09\} * s_{3,c}) \oplus (\{04\} * s_{3,c}) \oplus (\{02\} * s_{3c}) \oplus s_{3,c}] \quad (12)$$

From equation (9) to (12), it is clear that multiplication of source signals with {09}, {04} and {02} are redundantly used for calculation of Inv MixColumn. This common resources are represented are as follows,

$$c1_{09} = \{09\} * s_{0,c} \qquad c1_{04} = \{04\} * s_{0,c} \qquad c1_{02} = \{02\} * s_{0,c}$$

$$c2_{09} = \{09\} * s_{1,c} \qquad c2_{04} = \{04\} * s_{1,c} \qquad c2_{02} = \{02\} * s_{1,c}$$

$$c3_{09} = \{09\} * s_{2,c} \qquad c3_{04} = \{04\} * s_{2,c} \qquad c3_{02} = \{02\} * s_{2,c}$$

$$c4_{09} = \{09\} * s_{3,c} \qquad c4_{04} = \{09\} * s_{3,c} \qquad c4_{02} = \{02\} * s_{4,c}$$

The hardware complexity of Inv MixColumn can be absolutely reduced, when sharing these resources to all numerical calculation of Inv MixColumn. Further the equation (9) to (12) an be simplified as,

$$s'_{0,c} = (c1_{09} \oplus c1_{04} \oplus c1_{02} \oplus s_{0,c}) \oplus (c2_{09} \oplus c2_{02}) \oplus (c3_{09} \oplus c3_{02}) \oplus (c4_{09}) \quad (13)$$

$$s'_{1,c} = (c1_{09}) \oplus (c2_{09} \oplus c2_{04}) \oplus (c2_{02} \oplus s_{1,c}) \oplus (c3_{09} \oplus c3_{02}) \oplus (c4_{09} \oplus c4_{04}) \quad (14)$$

$$s'_{2,c} = (c1_{09} \oplus c1_{04}) \oplus (c2_{09}) \oplus (c3_{09} \oplus c3_{04} \oplus c3_{02} \oplus s_{2,c}) \oplus (c4_{09} \oplus c4_{02}) \quad (15)$$

$$s'_{3,c} = (c1_{09} \oplus c1_{04}) \oplus (c2_{09} \oplus c2_{04}) \oplus (c3_{09}) \oplus (c4_{09} \oplus c4_{04} \oplus c4_{02} \oplus s_{3,c}) \quad (16)$$

State bytes T_9 and T_4 are evaluated by as follows. Multiplication of {09} with state-byte,

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

Multiplication of {04} with state-byte,

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

6. Experimental Results

In this work, the enhanced inverse mix column with composite S-Box design is analyzed and evaluated in Modelsim XE by Verilog HDL Language and these synthesis results are estimated by Xilinx ISE Design suite. The hardware implementations of the AES encryption and decryption are implemented in Virtex-4 XC4VLX15 device through Xilinx ISE tool. Simulation result of Encryption by using Minimized Composite S-Box and Enhanced Inv Mix Column are shown in Fig.5 and Simulation result of Decryption by using Minimized Composite S-Box and Enhanced Inv Mix Column are shown in Fig.6

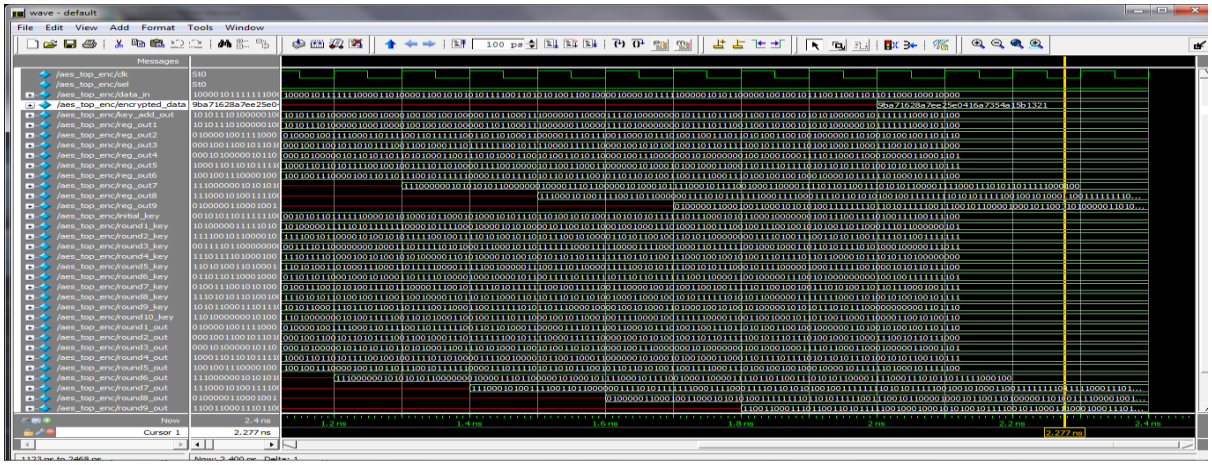

Fig.5 Simulation result of Encryption by using Minimized Composite S-Box and Enhanced Inv MixColumn

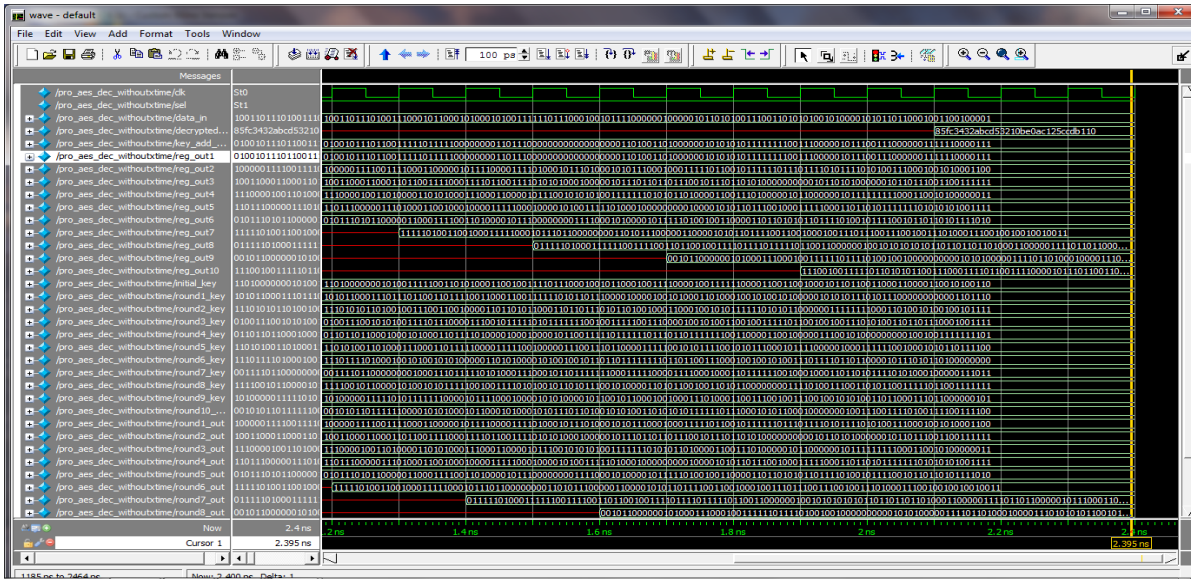

Fig.6 Simulation result of Decryption by using Minimized Composite S-Box and Enhanced Inv MixColumn

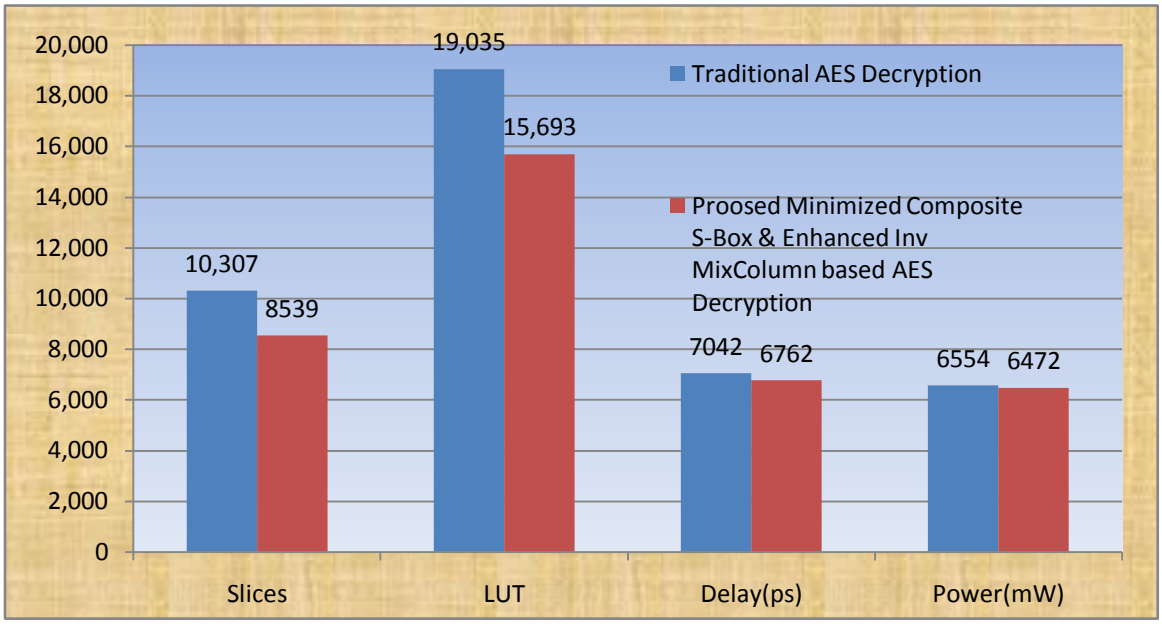

Fig.7 Comparisons for the existing and proposed AES Encryption and Decryption

7. Conclusion

In this work, the enhanced inverse mix column with composite S-Box design is simulated by hardware description language and these synthesis results are analyzed by Xilinx ISE. Proposed Composite S-Box and Enhanced Inv Mix Column transformations are implemented into AES encryption and AES decryption process respectively. The proposed design is provide 17% reduction of the slices and LUTs and 3% of the delay and efficient power consumptions. Proposed work performs for applications security Systems, Space and terrestrial communications.

REFERENCES

[1] Viktor Fischer, Milos Drutarovsky, Pawel Chodowiec and Francois Gramain, "InvMixColumn Decomposition and Multilevel Resource Sharing in AES Implementations" IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 13, No. 8, 2005.

[2] M. Anitha Christry and S. Sridevi Sathya Priya, "Design of Low Power Mixcolumn in Advanced Encryption Standard Algorithm" International Journal of Scientific & Engineering Research, Vol. 5, Issue. 4, pp: 64-68, 2014.

[3] Anitha, S., and Suganya, M., 2015. "Area optimized in storage area network using Novel Mix column Transformation in Masked AES" International Journal of Engineering Trends and Technology (IJETT), Vol. 20, No. 6, pp: 275-282.

[4] M. Senthil Kumar and S. Rajalakshmi, "Incorporation of Wave Pipelined Techniques into Composite S-Box and AES Architectures" *Research Journal of Applied Sciences, Engineering and Technology* (RJASET), Vol. 8, No. 15, pp: 1717-1723, 2014.

[5] Sandyarani, K., and Nirmal kumar, P., 2014. "Design of high speed AES-128 using Novel Mix Column Transformation and Sub Bytes" Journal of Computer Applications (JCA), Vol. 7, Issue. 2, pp: 57-60.