Implementation of AES with New S-Box and Performance Analysis with the Modified S-Box

K.Rahimunnisa
P.G scholar
Dept. of ECE, Karunya University

Dr. S. Sureshkumar
P.G scholar
Dept of EEE, Karunya University

K.Rajeshkumar
Professor & Head EEE department
Dept. of ECE, Karunya University

ABSTRACT
Cryptographic algorithms are the most essential elements in designing the system security. Though there are numerous encryption systems used in security systems by various organizations, for the wider use, a particular encryption method is used as a standard. The internationally accepted and acclaimed algorithm is Advanced Encryption Standard. Here in this paper we have implemented the Advanced Encryption Standard (AES) using hardware descriptive language (HDL) and constructed a modified S-box. Area and power of both S boxes are compared and their resistance to attack is also analysed.

General Terms
Security, Algorithms

Keywords
Encryption, AES, HDL,SBox.

1. INTRODUCTION
The National Institute of Standards and Technology (NIST) is a measurement standards laboratory of U.S federal agency. In order to establish Information Security, NIST sets the encryption standards under Federal Information Processing Standards (FIPS). On 26th November, 2001 NIST announced the Advanced Encryption Standard (AES) in FIPS 197 [1]. AES adopted the Rijndael Cipher, which was proposed by Joan Daemen and Vincent Rijmen as the algorithm for AES. Even till date AES is one of the most popular algorithms used in symmetric key cryptography.

AES uses the Rijndael encryption algorithm with cryptography keys of 128, 192, 256 bits. As in most of the symmetrical encryption algorithms, the AES algorithm manipulates the 128 bits of the input data, disposed in a 4 by 4 bytes matrix, with byte substitution, bit permutation and arithmetic operations in finite fields, more specifically, addition and multiplications in the Galois Field 2^8 ($GF(2^8)$). Each set of operations is designated by round. The round computation is repeated 10, 12 or 14 times depending on the size of the key (128, 192, 256 bits respectively). The coding process includes the manipulation of a 128-bit data block through a series of logical and arithmetic operations. In the computation of both the encryption and decryption, a well defined order exists for the several operations that have to be performed over the data block. [1]

2. RECENT RELATED WORKS
AES has been the standard Cryptographic algorithm commonly used across the universe. This is possible only because of its authenticity and its security. Though many research works have been done on this AES, very few focus on the Sbox of AES, which acts as the heart of the algorithm. Paper [5] presented a new mutable nonlinear transformation algorithm for AES S-box to enhance the complexity of the S-Box's structure called M_S-box .To improve the complexity of M_S-box' algorithm, the M_S-box structure has used a dynamic nonlinear transformation method and combined with linear function. The proposed M_S-box also uses statistic linear transformations.

The authors in paper [8] presented an overhead analysis for designing S-Box and GF(p) arithmetic blocks. They have used a heuristic gate as well as word-level synthesis and optimization technique for the analysis. Moreover, with regards to several performance index parameters, such as area, delay, and power a large set of experimental circuits has been designed. In paper [13] a detailed study on constructing a high throughput minimal power composite AES S-box was presented. They have utilized a balanced composite field AES S-box (in terms of area of implementation and critical path).In paper [10] the proposed S-Box gives another option for hardware implementation other than composite field to represent Subbyte transformation. It reduces the complexities of hardware by avoiding the use of multiplicative inverse in Galois field. As compared to the LUT and composite field, the S-Box resulted in smaller area with medium delay. In work [14], a novel heuristic common subexpression elimination (CSE) algorithm for binary linear transformation was presented. It was demonstrated that CSE algorithm is capable of reducing nearly half of the original complexity in the binary linear transformations of composite field AES S-box. The rest of the paper is organized as follows. Section 3 gives the working of AES and section 4 discusses the method to implement new S Box and linear cryptanalysis and section 5 gives the power and area analysis of both S Boxes.

3. METHODOLOGY
As for the AES algorithm, the length of the input block, the output block and the State is 128 bits. This is represented by N_b = 4, which reflects the number of 32-bit words (number of columns) in the State. For the AES algorithm, the length of the Cipher Key, K, is 128, 192, or 256 bits [5]. The key length is represented by N_k = 4, 6, or 8, which reflects the number of 32-bit words (number of columns) in the Cipher Key.

For the AES algorithm, the number of rounds to be performed during the execution of the algorithm is dependent on the key size. The number of rounds is represented by $\boldsymbol{N_r}$, where $\boldsymbol{N_r}$ = 10 when $\boldsymbol{N_k}$ = 4, $\boldsymbol{N_r}$ = 12 when $\boldsymbol{N_k}$ = 6, and $\boldsymbol{N_r}$ = 14 when $\boldsymbol{N_k}$ = 8. Increasing the key size increases the security level of the system, as the complexity in cryptanalysis will also increase.

TABLE I
Key-Block-Round Combinations

AES Version	Key Length (N_k words)	Block Size (N_b words)	Number of Rounds (N_r)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

The following describes in detail the operation performed by the AES encryption in each round. The State variable contains the 128-bit data block to be encrypted. In the Encryption part, first the data block to be encrypted is split into an array of bytes called as state matrix. This algorithm is based on round function, and different combinations of the algorithm are structured by repeating the round function different times. Each round function contains uniform and parallel four steps: SubBytes, ShiftRows, MixColumn and AddRoundKey transformation and each step has its own particular functionality [10]. The precomputation which is before the first round has one addround key. Similarly the N_r th round does not have the mix column operation.

3.1 Cipher

At the start of the Cipher, the input is copied to the State array. After an initial Round Key addition, the State array is transformed by implementing a round function 10, 12, or 14 times (depending on the key length), with the final round differing slightly from the first ***Nr*** -1 rounds. The final State is then copied to the output.

The round function is parameterized using a key schedule that consists of a one-dimensional array of four-byte words derived using the Key Expansion routine.

3.1.1 SubBytes Transformation

The SubBytes transformation is a non-linear byte substitution that operates independently on each byte of the State using a substitution table (S-box). This S-box, which is invertible, is constructed by composing two transformations:

1. Take the multiplicative inverse in the finite field GF(2^8), the element {00} is mapped to itself.
2. Apply the following affine transformation (over GF(2)):

$$b'_i = b_i \oplus b_{(i+4)\bmod 8} \oplus b_{(i+5)\bmod 8} \oplus b_{(i+6)\bmod 8} \oplus b_{(i+7)\bmod 8} \oplus c_i \quad (1)$$

for $0 < i < 8$ where b_i is the i_{th} bit of the byte, and c_i is the i_{th} bit of a byte c with the value {63} or {01100011}. (A prime on a variable indicates that the variable is to be updated with the value on the right).

3.1.2 ShiftRows Transformation

In the ShiftRows transformation, the bytes in the last three rows of the State are cyclically shifted over different numbers of bytes. The first row, $r = 0$, is not shifted. Specifically, the ShiftRows transformation proceeds as follows:

$$S'_{r,c} = S_{r,(c+shift(r,Nb))\bmod Nb} \quad (2)$$
$$\text{for } 0 < r < 4$$
$$\text{and } 0 \le c < N_b$$

where the shift value $shift(r,N_b)$ depends on the row number, r. This has the effect of moving bytes to "lower" positions in the row (i.e., lower values of c in a given row), while the "lowest" bytes wrap around into the "top" of the row (i.e., higher values of c in a given row).

3.1.3 MixColumns Transformation

The MixColumns transformation operates on the State column-by-column, treating each column as a four-term polynomial. The columns are considered as polynomials over GF(2^8) and multiplied modulo $x^4 + 1$ with a fixed polynomial $a(x)$, given by

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} \quad (3)$$

this can be written as a matrix multiplication.

3.1.4 AddRoundKey Transformation

In the AddRoundKey transformation, a Round Key is added to the State by a simple bitwise XOR operation. Each Round Key consists of N_b words from the key schedule. Those N_b words are each added into the columns of the State, such that

$$[s'_{0,c}, s'_{1,c}, s'_{2,c}, s'_{3,c}] \oplus [w_{round \cdot N_b + c}] \quad \text{for } 0 < c < N_b \quad (4)$$

where [*wi*] are the key schedule words described below, and *round* is a value in the range $0 \le round \le Nr$. In the Cipher, the initial Round Key addition occurs when *round* = 0, prior to the first application of the round function. The application of the AddRoundKey transformation to the ***Nr*** rounds of the Cipher occurs when $1 \le round \le Nr$.

3.1.5 Key Expansion

The AES algorithm takes the Cipher Key, ***K***, and performs a Key Expansion routine to generate a key schedule. The Key Expansion generates a total of $N_b (N_r + 1)$ words: the algorithm requires an initial set of N_b words, and each of the N_r rounds requires N_b words of key data. The resulting key schedule consists of a linear array of 4-byte words, denoted [*wi*], with *i* in the range $0 < i < N_b(N_r + 1)$ [13]

4. CREATION OF NEW S BOX

Based on the various conditions to design a S-box[9], designing an Sbox is difficult and the security analysis of Sbox becomes more complex. These variations are designed over the Galois field GF(2^8) generated by the primitive polynomial

$$x^8 + x^6 + x^5 + x + 1$$

Based on the work done by V. Ch. Venkaiah and Srinathan Kannan a new Sbox was designed. The design steps are as follows.

It is designed by composing two transformations:[11]

1. Power of a primitive element in F^*_{257}. If the power of the chosen primitive element is 256, it is treated as 00.
2. Multiplicative inverse in the finite field GF(2^8). Element 00 is mapped to itself.

A way of realizing this transformation by choosing 3 as the primitive element in F^*_{257} is as follows:

1. Initialize the S – box with the integer values in ascending sequence row by row. The first row contains {00}, {01}, {02}, …,{0F}; the second row contains {10}, {11}, etc.; and so on. Thus the value of the byte at row x and column y is {xy}.
2. Map each byte {xy} in the S – box to an element $3^{\{xy\}}$ mod 257, an element in F^*_{257}. For example, if {xy} = {32} then according to this map {32} gets mapped to $3^{\{32\}}$ mod 257 = 3^{50} mod 257 = 18 (decimal) = {12}. But $3^{\{80\}}$ mod 257 = 3^{128} mod 257 = 256 (decimal) is treated as {00}[15].

3. Map each resulting byte in the S – box to its multiplicative inverse in the finite field GF(2^8). {00} gets mapped to itself.

Steps 1 and 3 may be easily identified with the steps 1 and 2 of S – box construction in [1]. These steps lead to the construction of a new S Box which is different from the standard S Box.

5. ANALYSIS OF S-BOX

Linear cryptanalysis tries to take advantage of high probability occurrences of linear expressions involving plaintext bits, "cipher text" bits (actually we shall use bits from the 2nd last round output), and sub key bits.
Based on A Tutorial on Linear and Differential Cryptanalysis by Howard M. Heys, a simple linear and cryptanalysis is done on the standard S box and the new one as discussed below.
The basic idea is to approximate the operation of a portion of the cipher with an expression that is linear where the linearity refers to a mod-2 bit-wise operation (i.e., exclusive-OR denoted by "⊕"). Such an expression is of the form:.

$$X_{j1} \oplus X_{j2} \oplus \ldots \oplus X_{ju} \oplus Y_{j1} \oplus Y_{j2} \ldots\ldots \oplus Y_{jv} = 0 \qquad (5)$$

Where *Xi* represents the *i*-th bit of the input $X = [X_1, X_2, ...]$ and *Yj* represents the *j*-th bit of the output $Y = [Y_1, Y_2, ...]$.

If randomly selected values for $u + v$ bits are placed into the equation above, the probability that the expression would hold would be exactly 1/2. The further away that a linear expression is from holding with a probability of ½ (linear probability bias), the better the cryptanalyst is able to apply linear cryptanalysis.

If the expression above holds with probability p_L for randomly chosen plaintexts and the corresponding ciphertexts, then the probability bias is $p_L - 1/2$. The higher the magnitude of the probability bias, $|p_L - 1/2|$, the better the applicability of linear cryptanalysis with fewer known plaintexts required in the attack.

The probability that $X_1 \oplus ... \oplus X_n = 0$ holds can be determined by the *Piling-Up Lemma* which assumes that all *n* random binary variables are independent.

For *n* independent, random binary variables, $X_1, X_2, ...X_n$,

$$P_r(X_1 \oplus \ldots \oplus X_N = 0) = \tfrac{1}{2} + 2^{n-1} \prod_{i=1}^{n} \varepsilon i \qquad (6)$$

or

$$\varepsilon_{1,2,\ldots,n} = 2^{n-1} \prod_{i=1}^{n} \varepsilon_i \qquad (7)$$

where $\varepsilon_{1,2,...,n}$ represents the bias of $X_1 \oplus ... \oplus X_n = 0$.

Note that if $p_i = 0$ or 1 for all *i*, then $\Pr(X_1 \oplus ... \oplus X_n = 0) = 0$ or 1. If only one $p_i = 1/2$, then $\Pr(X_1 \oplus ... \oplus X_n = 0) = 1/2$. In developing the linear approximation of a cipher, the X_i values will actually represent linear approximations of the S-boxes.

5.1 Area Comparison

Area of the standard S Box and Modified S Box are calculated using synopsys Design Vision tool. Area is plotted in Fig 4.

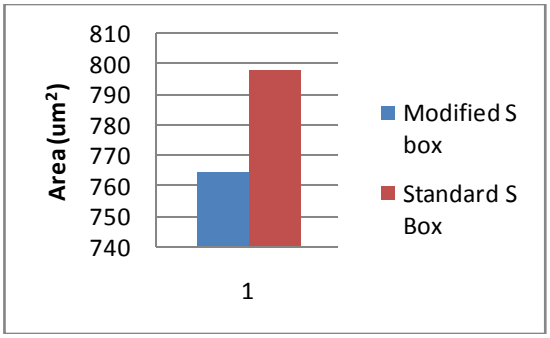

Figure 4. Area comparison of standard and modified S-Box.

Fig. 4 gives the area comparison of the two S-boxes. The area of the normal and the proposed S-Box has been compared and it has been seen that the normal S-Box comprises of 16 ports, 498nets, 490 cells and 21 references as a total. It has area occupied as 798 um^2 and in the modified S-Box it has been seen that it has 16 ports, 478 nets, 470 cells and as a total it has total cell area of 765 um^2

5.2 Power Comparison

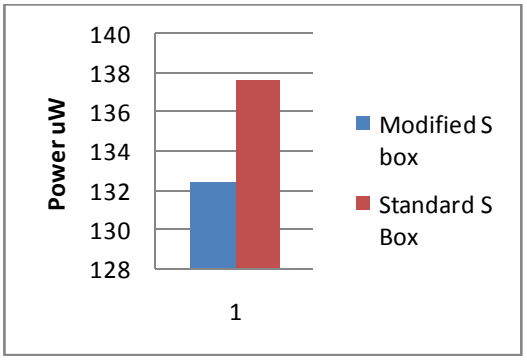

Figure 5. Power Comparison of Standard and Modified S-Box

Fig 5 shows the power comparison between normal S-box and modified S-box. The Net Switching power and dynamic power of S-Box is 137.6804 uW and the modified S-Box has 132.4501 uW . From the figure it is clear that the modified S-box utilizes less power.

5.3 Security Analysis

From (5) and (6) the linear expression for the standard S – Box is

$$X_7 \oplus Y_2 \oplus Y_3 \oplus Y_4 \oplus Y_5 = 0 \text{ or equivalently}$$
$$X_7 = Y_2 \oplus Y_3 \oplus Y_4 \oplus Y_5$$

Applying all 256 possible input values for *X* and examining the corresponding output values *Y*, it may be observed that for exactly 127 out the 256 cases, the expression above holds true. Hence, the probability bias is 127/256−1/2 = -1/256. In this case, the best approximation is an affine approximation as indicated by the minus sign. However, the success of the attack is based on magnitude of the bias.

Similarly from (5) and (6) the linear expression for modified S-Box is
$X_7 \oplus Y_0 \oplus Y_1 \oplus Y_3 \oplus Y_4 \oplus Y_5 \oplus Y_6 = 0$ or equivalently
$X_7 = Y_0 \oplus Y_1 \oplus Y_3 \oplus Y_4 \oplus Y_5 \oplus Y_6$

Applying all 256 possible input values for *X* and examining the corresponding output values *Y*, it may be observed that for exactly 128 out the 256 cases, the expression above holds true. Hence, the probability bias is 127/256–1/2 = -1/256. In this case, the best approximation is an affine approximation as indicated by the minus sign.

7. CONCLUSIONS

The standard S Box and the modified S Box are developed and prototyped in an FPGA platform using the Verilog® Hardware Description Language (HDL) and the Spartan 2E. The synthesis is done using HDL Designer (Mentor Graphics). The simulation shows that this design can run at 1.2GHz which is sufficient for online data encryption. However, it has been seen that both the probability bias are same. Therefore with same security complexity the area and power are improved marginally. Analysis of area and power was calculated through Design Vision (Synopsys). More work is to be done on the security analysis.

8. REFERENCES

[1] AES page http://www.nist.gov/CryptoToolkit.4.

[2] A.Lee. 1999. Guideline for Implementing Cryptography in the Federal Government. NIST Special Publication 800-21. National Institute of Standards and Technology.

[3] B. Gladman's AES related home page http://fp.gladman.plus.com/cryptography_technology/.

[4] J. Daemen and V. Rijmen. 1999. AES Proposal: Rijndael, AES Algorithm.

[5] Hiroshi Kudou, Shunn-Ichiro Nakayama, Atsushi Watanabe, Tomoyuki Nagase , Yoshio Yoshioka. 2009. A Reconfigurable-Permutation Algorithm For M_S-Box. IEEE Computer Society. International Conferenceon Availability, Reliability and Security.

[6] Mao-Yin Wang, Chih-Pin Su, Chia-Lung Horng, Cheng-Wen Wu, And Chih-Tsun Huang. 2010. Single- and Multi-core Configurable AES Architectures for Flexible Security. IEEE Transactions On Very Large Scale Integration (VLSI) Systems. Vol. 18. No. 4. 541-552.

[7] Mehran Mozaffari-Kerman, and Arash Reyhani-Masoleh. 2011. A Lightweight High-Performance Fault Detection Scheme for the Advanced Encryption Standard Using Composite Fields. IEEE Transactions On Very Large Scale Integration (VLSI) Systems. Vol. 19. No. 1. 85-91.

[8] J. Mathew, H. Rahaman, A. M. Jabir, S. P. Mohanty and Dhiraj K. Pradhan. 2010. On the Design of Different Concurrent EDC Schemes for S-Box and GF(*p*). 11th Int'l Symposium on Quality Electronic Design. 11th Int'l Symposium on Quality Electronic Design.

[9] William Stallings. 2004. Cryptography and Network Security . Pearson education.

[10] Zine El Abidine Alaoui Ismaili, Ahmed Moussa. 2009. Self-Partial and Dynamic Reconfiguration Implementation for AES using FPGA. IJCSI International Journal of Computer Science. Vol. 2. 33-40.

[11] V. Ch. Venkaiah and Srinathan Kannan. 2006. IIIT Hyderabad Publications.

[12] Howard M. Heys. A Tutorial on Linear and Differential Cryptanalysis.

[13] M. M. Wong and M. L. D. Wong. 2010. A High Throughput Low Power Compact AES S-box Implementation using Composite Field Arithmetic and Algebraic Normal Form Representation. 2nd Asia Symposium on Quality Electronic Design.

[14] M. M. Wong and M. L. D. Wong. 2010. A New Common Subexpression Elimination Algorithm with Application in Composite Field AES S-BOX. 10th International Conference on Information Science, Signal Processing and their Applications .

[15] Nabihah Ahmad ,Rezaul Hasan,Warsuzarina Mat Jubadi. 2010. Design of AES S-Box using Combinational logic optimization. IEEE Symposium on Industrial Electronics and Applications.