Enhanced Key Expansion Algorithm for Advanced Encryption Standard using Different S- Box Implementation on FPGA

Amrutha T V
PG Scholar
Department of Electronics & Communication Engineering
KIT, Tiptur, India

N R Prashanth
Associate Professor
Department of Electronics & Communication Engineering
KIT, Tiptur, India

Abstract

The main aim of this paper is encrypt the data using Advanced Encryption Standard (AES) algorithm. In AES algorithm cryptography technique is used. Security is most important in data communication so to increase the security key expansion algorithm is used. In this paper we considering different sizes of S-box to reduce the area and the LUTs. To reduce LUTs here considering the affine transformation method is used. The round key expansion is proposed to improve security against attacks. Encrypted data is decrypted using inverse AES algorithm method. In AES algorithm numbers of round performed during execution will be depended upon the Key length. Here AES -128-bit key are used, so number of round performed during execution will be 10. This algorithm is simulated using Xilinx software and implemented on FPGA.

Keywords- Advanced Encryption Standard; FPGA; Key expansion; Sub word; Rotword; Rconst

I. INTRODUCTION

Cryptography assumes a vital part in securing the data, which empowers to store delicate data or transmit it crosswise over unreliable systems so that unknown persons can't get it. The Advanced Encryption Standard (AES) was distributed by the National Institute of Standards and Technology (NIST) in 2001. AES is a cryptographic calculation that is utilized to secure electronic information and it has symmetric block cipher with a piece length of 128 bits that can encipher (scramble) the information. The AES calculation comprises of three fundamental parts:

1) cipher (Encryption),
2) inverse Cipher (Decryption)
3) Key Expansion.

Encryption changes over information to a mixed up structure called as cipher content. decryption changes over the cipher content once again into its indigenous structure that is original information. The AES calculation is equipped for utilizing diverse cryptographic key lengths of 128, 192, and 256 bits to encipher and decipher the information. Key development is utilized for creating the keys for 10 adjusts that are delivered from the first data key. Each round key are not the same as each other to enhance the security of the calculation. In this manner it inessential to enhance the Performance of the key development from the exteral attacks. But the tradition key extension of AES has some security issue because of the key courses of action are relying on the past key rounds. Thus, once again calculation of key development is proposed to enhance the security of the Advanced Encryption Standard for 128-piece key size.

II. ADVANCED ENCRYPTION STANDARD ALGORITHM

A. AES Specification

The length of the info hinders, the yield square and the State is 128 bits for the AES calculation which is spoken to by Nb = 4. The information 128-bits are organized in 4 × 4 grid into16 bytes that mirrors the quantity of 32-bit words (number of segments) in the State. The AES calculation will bolster in any event of the three key lengths: 128, 192, or 256-bits (i.e., Nk = 4,6, or 8, individually). The length of key is spoken to by Nk =4, 6, or 8 which mirrors the quantity of 32bit words (number of segments) in the Cipher Key. The quantity of rounds for ASE calculation tbe performed amid the execution is reliant on the key size. The quantity of rounds is spoken to by Nr, where N10 when Nk = 4, Nr = 12 when Nk = 6, and Nr = 14when Nk = 8**.** AES calculation utilizes a round capacity for both its Cipher and Inverse Cipher that is made out of four distinctive byte arranged changes:

- Byte substitution utilizing a S-box lookup table.
- Row-wise stage of the State cluster by various balances
- Column-wise blending inside every section of the State exhibit
- Addition of round key to the State.

The structure of AES calculation for both the encryption and unscrambling is appeared in the Fig.1.which demonstrates the general procedure.

B. Encryption Process

The initial procedure in AES encryption is the expansion exoring of original key to the information, which is called an initial round. This is trailed by nine iterations of an ordinary round and end with an altered last round. During every ordinary round the following operations are performed in the accompanying request: Byte substitution, Row-wise change, Column-wise blending, Addition of the round key. The last round is additionally an ordinary round without the Column-wise blending process

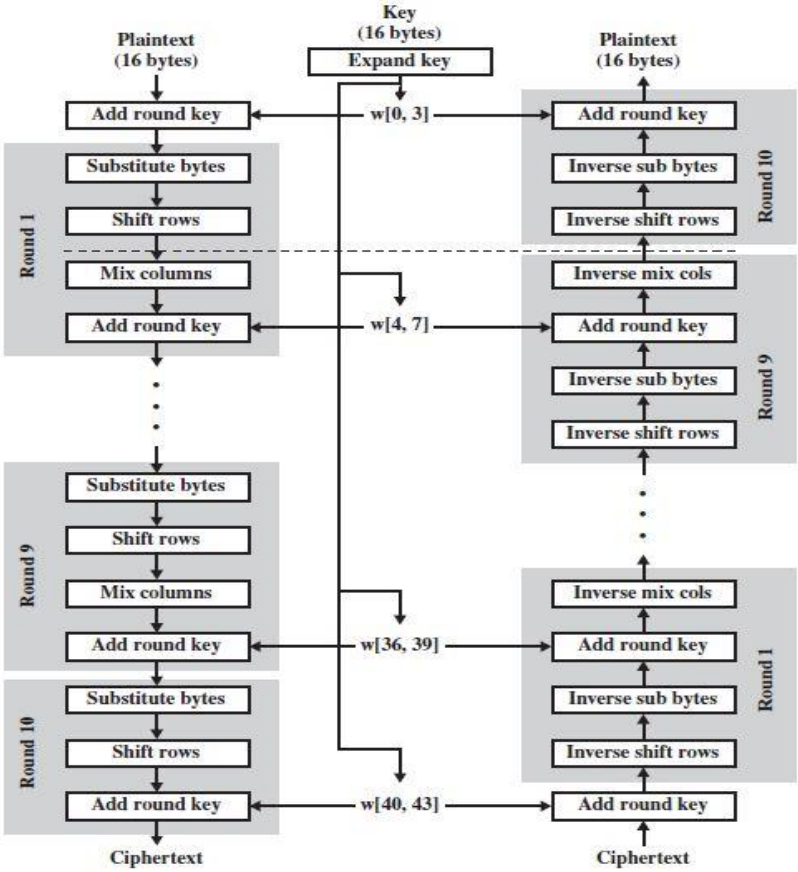

Fig. 1: Structure of AES

1) Byte Substitution

This is a byte-by-byte substitution process appeared in Fig. 2. Which forms substitution of bytes. The substitution byte for every data byte is found by utilizing the S-Box lookup table. The extent of the lookup table is 16×16. To locate the substitute byte for a given data byte, we isolate the information byte into two 4-bit designs, every yielding a whole number quality somewhere around 0 and 15 which can speak to these by hex qualities 0 through F. One of the hex qualities is utilized as a line file and alternate as a segment file for venturing into the16×16 lookup table.

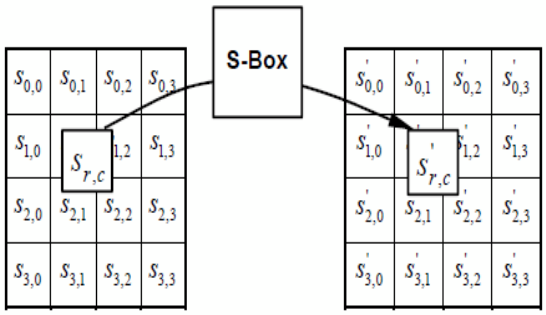

Fig. 2: Byte Substitution.

The passages in the lookup table are built by blend of GF (28) math and bit scrambling. The objective of the substitution step is to decrease the connection between the information bits and the yield bits. The bit scrambling part of the substitution step guarantees that the substitution cannot be depicted through assessing a straightforward numerical capacity.

2) Row-Wise Permutation

The Row-wise change comprises of (i) not moving the main line of testate cluster by any means (ii) circularly moving the second column by one byte to one side (iii) circularly moving the third line by two bytes to one side and (iv) circularly moving the last line by three bytes to one side. This operation on the state cluster can be spoken to b

$$\begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix} \quad ====> \quad \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,1} & S_{1,2} & S_{1,3} & S_{1,0} \\ S_{2,2} & S_{2,3} & S_{2,0} & S_{2,1} \\ S_{3,3} & S_{3,0} & S_{3,1} & S_{3,2} \end{bmatrix}$$

3) Column-Wise Mixing

This stride replaces every byte of a section by a component of the considerable number of bytes in the same segment. For the bytes in the primary column of the state, operation can be expressed as

S'0,c = ({02} • S0,c) + ({03} • S1,c) + S2,c + S3,c (1)

For the bytes in the second line of the state can be expressed as

S'1,c = S0,c + ({02} • S1,c) + ({03} • S2,c) + S3,c (2)

For the bytes in the third line of the state can be expressed as

S'2,c = S0,c + S1,c + ({02} • S2,c) + ({03} • S3,c) (3)

What's more, for the bytes in the fourth column of the state exhibit can be expressed as

S'3,c = ({03} • S0,c) + S1,c + S2,c + ({02} • S3,c) (4)

All the more minimalistic ally, the section operations can be appeared as Where, a line of the furthest left lattice products a section of the state exhibit framework, augmentations included are intended to be XOR operation.

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \times \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix} = \begin{bmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{2,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{3,3} \end{bmatrix}$$

4) Addition of Round Key

The watchwords are produced by key extension process. Round Key is added to each State by a XOR operation where each Round Key comprises of Nb words. Those Nb words are each additional into the segments of the State, such that [wi] are the key calendar words and round is a worth in the extent 0 round Nr. In the Cipher, the underlying Round Key expansion happens when round = 0, preceding the main use of the round capacity. Add Round Key change to the Nr rounds of the Cipher happens when 1<round <Nr. A straightforward xor operation with the state to catchphrase is appeared in Fig. 3.

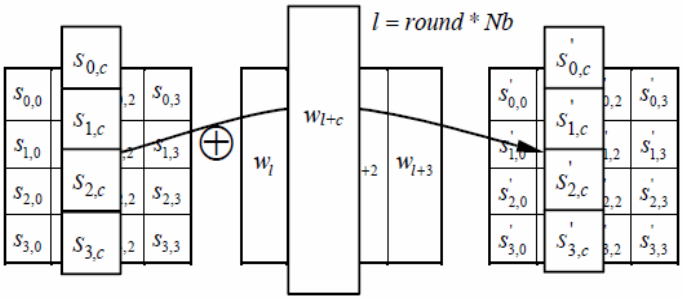

Fig. 3: Addition of Round Key XORs each column.

C. Decryption Process

In AES decoding, the operations are backward request. It begins with an underlying round took after by nine cycles of a reverse round and closes with an Addition of Round Key. A converse round comprises of the accompanying operations in a specific order: Addition of Round Key, Inverse segment shrewd blending, Inverse Row-wise stage and Inverse Substitution of Bytes. An underlying round is a converse round without the Inverse Column shrewd blending and expansion of round key has its own reverse for both the encryption and decoding process.

1) Inverse Row-Wise Change

For unscrambling, the comparing step moves the lines in inverse way. The main line is left unaltered, the second column is moved to one side by one byte, the third line to one side by two bytes, and the last line to one side by three bytes, all are circularly moved.

$$\begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix} \quad ====> \quad \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,3} & S_{1,0} & S_{1,1} & S_{1,2} \\ S_{2,2} & S_{2,3} & S_{2,0} & S_{2,1} \\ S_{3,1} & S_{3,2} & S_{3,3} & S_{3,0} \end{bmatrix}$$

2) Inverse Column Wise Change

It is the backwards procedure of segment astute blending that works on the State segment by segment and taking every segment as a four term polynomial. The sections are considered as polynomials over GF (28) and duplicated modulo $x^4 + 1$ with a settled polynomial a-1(x), given by a-1(x) = {0b} x^3 + {0d} x^2 + {09} x + {0e}, this can be composed as a grid augmentation.

3) Inverse Substitution of Bytes

The converse substitution byte for every information byte is found by utilizing the backwards s-box lookup table. The span of the lookup table is 16×16. To locate the opposite substitution byte for a given information byte, we separate the data byte into two 4-bit designs, every yielding a whole number quality somewhere around 0 and 15 which can speak to these by hex qualities 0 through F. One of the hex qualities is utilized as a line record and alternate as a segment list for venturing into the16×16 lookup table. The passages in the lookup table are built by a multiplicative converse of GF.

III. PROPOSED KEY EXPANSION ALGORITHM

It is crucial to improve the security of a cryptographic calculation in information correspondence. In Advanced Encryption Standard, the customary key development calculation has some security issues because of the deducible key game plans. It is known the 4-expressions of round key in each round can be gotten from the 4-expressions of previous round key. Despite what might be expected, the 4-expressions of previous round key can likewise be concluded by the last ones.

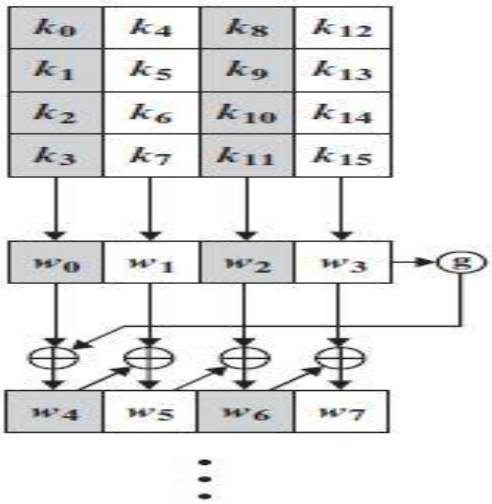

Fig. 4: Key Expansion process.

Thus, in the event that one round key is known, the aggressor can derive the sub-key of each round and even the seed key. In the interim, Power examination assault and Saturation assault which are viable to AES, both make utilization of the basic watchword game plan of key extension calculation. Subsequently, the Advanced Encryption Standard with another calculation of improved key development game plan is proposed. All together not to make the key producing calculation more entangled, it is important to confuse the operations. The creating calculation is as per the following

– wi = wi-8 + Subword(Rotword(wi-4))+Rcon(i/4)

Where I =8,12… ..40

– wi = wi-8 + wi-4 ; (8<i<44 and i is not a different of 4)

The sub-watchwords of the first round, it can be just produced from the first key. It has "one way" character so induction must be done from previous to later.

– w4= w0 + w2
– w5= w1 + w3
– w6= w4 + w5
– w7= w5 + Sub word (Rotword (w6))+Rcon(1)

From the second round, the key development procedure will be same as
Catches up to tenth round.

– w8= w0 + w4
– w9= w1 + w5
– w10= w2 + w6
– w11= w3 + Sub Word(Rotword(w7))+Rcon(2)

Assuming assailants have known the key (w4, w5, w6, w7), it is difficult to conclude the first key (w0, w1, w2, w3) , in light of the fact that w7only relies on upon w5 and w6, w6 just relies on upon w4 andw5,while w5 depends just on w1 and w3. Regardless of the possibility that w5 known, 232exhaustive assaults need to get w1 and w3 (every character length is 32bit). For the same reason, to get w0 and w2 from w4, 232times assaults are required. In this way, to get the first round sub-key, still they require 264 times to figure the first key. As indicated by the investigation over, the aggressor needs to split two progressive rounds of sub-watchwords to get the entire key bits. So the proposed calculation for key development has higher key security contrasted with the conventional AES key extension calculation, however the many-sided quality continues as before.

A. Modular Inversion in a Composite Field

The most scientifically complex operation of the AES figure is secluded reversal in a limited field. A more minimal execution of the figure's S-box can be gotten by performing particular reversal in the field $GF(2^4)^2$ instead of in $GF(2^8)$. The field over $GF(2^8)$ called an expanded field while the more minimal representation $GF(2^4)^2$ is known as a composite field. The field $GF(2^4)$ is alluded to as a subfield. To perform secluded reversal in the amplified field requires a 256 byte LUT.

A proportionate representation of the broadened Rijndael field into a more packed subfield over GF(24)2 diminishes the span of the LUT to only 8 bytes. represents the outline stream of SubBytes for Rijmen's productive S-box execution for encryption demonstrates the procedure for decoding Every component of $GF(2^8)$ can be composed as a polynomial of the primary degree with coefficients from $GF(2^4)$. This polynomial has the structure bx + c, where b and c are coefficients from $GF(2^4)$ **)** ..

IV. Simulation Result Of AES Encryption And Decryption Using proposed Key Expansion Algorithm

16-byte information is reenacted utilizing the Xilinx programming instrument. The encryption procedure of the AES calculation produces a cipher key for a given information**.**

A. Encryption

In the encryption prepare the AES calculation produces a cipher key for a given 16-byte information and figure key. The reproduction result is appeared in Fig5.

Input = [00112233445566778899aabbccddeeff]
Key = [000102030405060708090a0b0c0d0e0f]
CT = [69c4e0d86a7b0430d8cdb78070b4c55a]

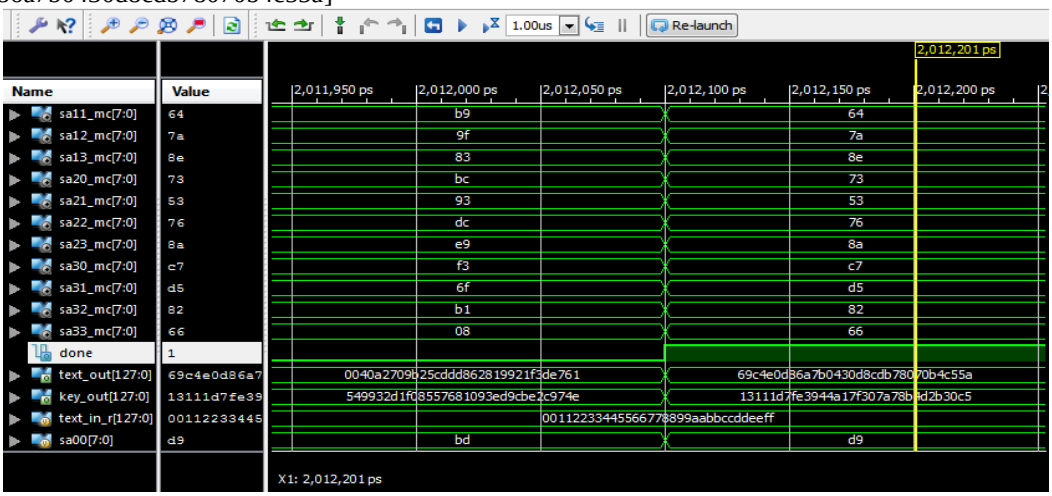

Fig. 5: Encryption Simulation Result

B. Decoding

The figure content acquired in the encryption procedure is given as the information of unscrambling procedure and the first info information is gotten by decoding. The reproduction result appeared in Fig6.

Input text = [69c4e0d86a7b0430d8cdb78070b4c55a]

Key = [000102030405060708090a0b0c0d0e0f]

Original key = [00112233445566778899aabbccddeeff]

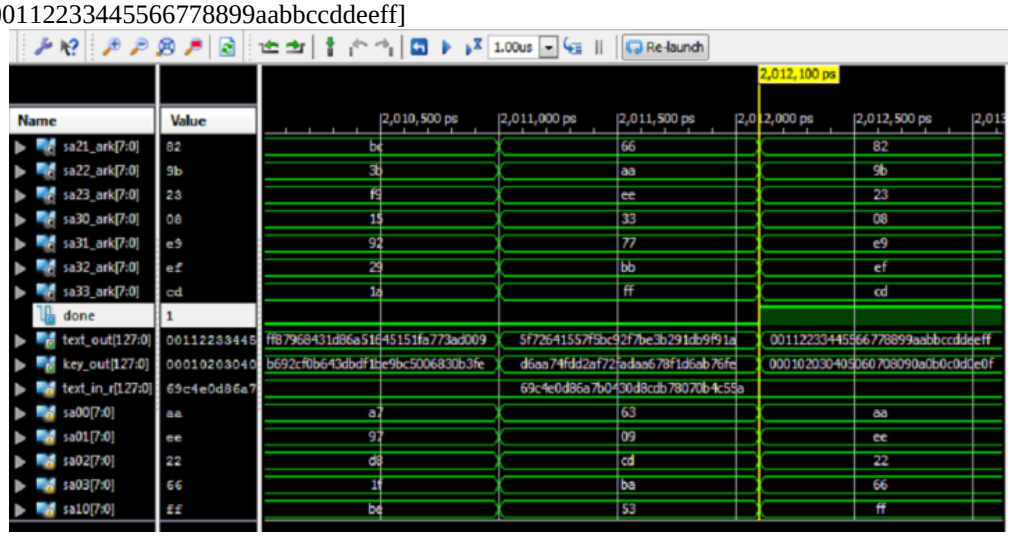

Fig. 6: Decryption Simulation Result

V. CONCLUSION

The point of this proposed calculation is to expand the level of security for Advanced Encryption Standard and giving a speedier handling time. Also, the trouble of assaulting the key is expanded which controls the capacity to oppose significant assaults because of the high randomization of key development in the new calculation. The synthesized using XILINX 14.5 and executed on FPGA. The simulation result for AES-128 is obtained by simulating the proposed key expansion algorithm using Verilog Hardware Description Language.

REFERENCES

[1] AI-Wen Luo, Qing-Ming Yi, Min Shi. "Design and Implementation of Area-optimized AES on FPGA", IEEE Inter. conf. chal sci comengin.,978-1-61284-109-0/2011.
[2] H.Mestiri, N.Benhadjyoussef, M.Machh out and R.Tourki, "A Comparative Study of Power Consumption Models for CPA Attack,"
[3] International Journal of Computer Network and Information Security, Vol. 5, No. 3, pp.25-31, 2013.
[4] A. Moh'd, Y.Jararweh and L. Tawalbeh, "AES-512: 512-bit Advanced Encryption Standard algorithm design and evaluation," 7th International Conference on Information Assurance and Security (IAS 2011), pp. 292-297, 2011.
[5] M. Mozaffari-Kermani, and A. Reyhani-Masoleh, "Concurrent structure independent fault detection schemes for the advanced encryption standard," IEEE Transactions on Computers, Vol. 59, pp.608-622, 2010.
[6] H. Mestiri, N. Benhadjyoussef, M. Machhout and R. Tourki, "A Comparative Study of Power Consumption Models for CPA Attack," International Journal of Computer Network and Information Security,Vol. 5, No. 3, pp. 25-31, 2013.