UACEE International Journal of Advances in Electronics Engineering
Vol:1 Issue:1 ISSN 2278 - 215X

Design of AES Algorithm using FPGA

Atul M. Borkar
Student [M-Tech (VLSI)]
Department of Electronics Engineering
Priyadarshini College of Engineering,
Nagpur, Maharashtra, India.
e-mail: borkar.atul@gmail.com

Dr. R. V. Kshirsagar
Vice Principal
Department of Electronics Engineering
Priyadarshini College of Engineering,
Nagpur, Maharashtra, India.

Mrs. M. V. Vyawahare
Lecturer
Department of Electronics Engineering
Priyadarshini College of Engineering,
Nagpur, Maharashtra, India.

Abstract—The Advanced Encryption Standard can be programmed in software or built with pure hardware. However Field Programmable Gate Arrays (FPGAs) offer a quicker, more customizable solution. This research investigates the AES algorithm with regard to FPGA and the Very High Speed Integrated Circuit Hardware Description language (VHDL). Software is used for simulation and optimization of the synthesizable VHDL code. All the transformations of both Encryptions and Decryption are simulated using an iterative design approach in order to minimize the hardware consumption.

Keywords: AES algorithm (encryption, decryption), key expansion, hardware implementation.

I. INTRODUCTION

The National Institute of Standards and Technology, (NIST), solicited proposals for the Advanced Encryption Standard, (AES). The AES is a Federal Information Processing Standard, (FIPS), which is a cryptographic algorithm that is used to protect electronic data. The AES algorithm is a symmetric block cipher that can encrypt, (encipher), and decrypt, (decipher), information. Encryption converts data to an unintelligible form called cipher-text. Decryption of the cipher-text converts the data back into its original form, which is called plaintext. The AES algorithm is capable of using cryptographic keys of 128, 192, and 256 bits to encrypt and decrypt data in blocks of bits.

Cryptography plays an important role in the security of data. It enables us to store sensitive information or transmit it across insecure networks so that unauthorized persons cannot read it. The urgency for secure ex-change of digital data resulted in large quantities of different encryption algorithms which can be classified into two groups: asymmetric encryption algorithms (with public key algorithms) and symmetric encryption algorithms (with private key algorithms). Symmetric key algorithms are in general much faster to execute electronically than asymmetric key algorithms.

II. DESCRIPTION OF AES ALGORITHM

The algorithm is composed of three main parts: Cipher, Inverse Cipher and Key Expansion. Cipher converts data to an unintelligible form called ciphertext while Inverse Cipher converts data back into its original form called plaintext. Key Expansion generates a Key Schedule that is used in Cipher and Inverse Cipher procedure. Cipher and Inverse Cipher are composed of specific number of rounds (Table 1). For the AES algorithm, the number of rounds to be performed during the execution of the algorithm is dependent on the key length.

Table 1: Different keys and its attributes

Algorithm	Block Size (Nb Words)	Key Length (Nk Words)	Number of Rounds (Nr)
AES–128-bits key	4	4	10
AES–192-bits key	4	6	12
AES–256-bits key	4	8	14

For encryption and decryption there are 4 different steps

Sub Bytes: Every byte in the state is replaced by another one using S-box (Substitution Box).

Shift Rows: Every row in the state (4×4 array) is shifted left k bytes and the k depends on the key and the row number.

Mix Column: A linear mixing operation which operates on the columns of the state, combining the four bytes in each column.

Add Round key: Each byte of the state is combined with a round key, which is different key for each round.

The sequence in which the operation is carried out is as follows:

Round 1:

A. Add Round key.

Following Rounds:

A. Sub Bytes.
B. Shift Rows.
C. Mix Column.
D. Add Round Key.

Final Round:

A. Sub Bytes.
B. Shift Row.
C. Add Round Key.

This is shown in figure 1. The AES algorithm can be implemented in both hardware and software. The software implementation of AES algorithm is a slow process when compared with hardware process.

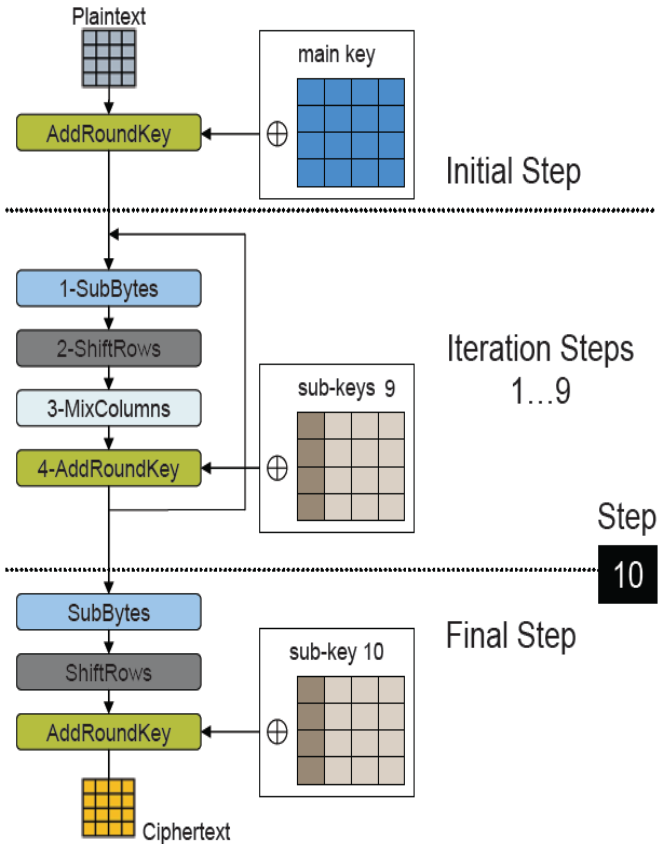

Figure 1: Basic Concept of the Algorithm

A. AES Encryption Process:

Encryption is the process of converting the plain text into a format which is not easily readable and is called as cipher. The cipher is got by doing a series of mathematical operations iteratively.

a) Sub Bytes:

In this sub bytes step the data in the plain text is substituted by some pre-defined values from a substitution box. The substitution box is invertible.

b) Shift Rows:

In shift rows operation the rows in the 4×4 matrix is shifted to left r bits and r varies with the rows of the matrix(r=0 for row1, r=1 for row2, r=2 for row3, r=3 for row 4). This process is illustrated in figure 2.

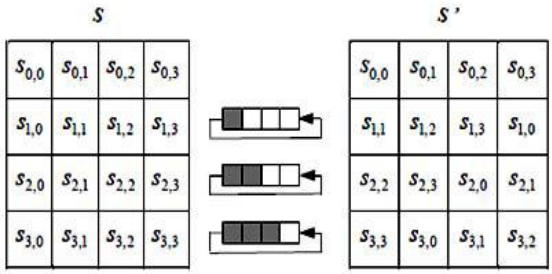

Figure 2: Shift Rows

This has the effect of moving positions of lower positions in the row, while the lowest bytes wrap around to the top of the row.

c) Mix Columns:

Mix column is calculated using the below formula.

$$\begin{bmatrix} R0 \\ R1 \\ R2 \\ R3 \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} a0 \\ a1 \\ a2 \\ a3 \end{bmatrix}$$

Here a0, a1, a2, a3 is calculated using the polynomials as below

$$a(x) = \{2\}x^3 + \{3\}x^2 + \{1\}x + \{1\}.$$

The mix column 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)** which is got from the above formula. This can also written as a matrix multiplication

$$s'(x) = a(x) \otimes c(x)$$

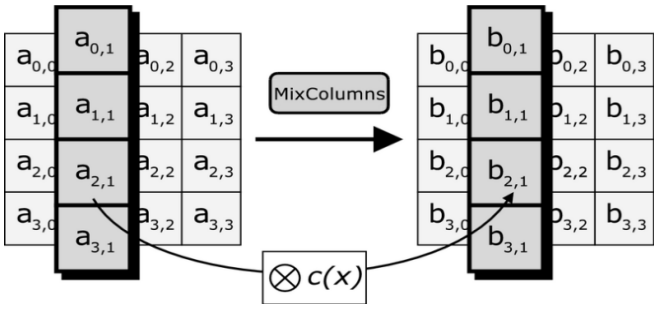

Figure 3: Mix Column

d) Add Round Key:

In the add round key step the 128 bit data is xored with the sub key of the current round using the key expansion operation. The add round key is used in two different places one during the start that is when round r=0 and then during the other rounds that is when 1 ≤ round≤ Nr, where Nr is the maximum number of rounds. The formula to perform the add round key is

$$S'(x) = S(x) \oplus R(x)$$

S'(x) – state after adding round key
S(x) – state before adding round key
R(x) – round key

e) Key Expansion:

The key expansion has three steps:

a) Byte Substitution *subword()*
b) Rotation *rotword()*
c) Xor with RCON (round constant)

The input to key schedule is the cipher key K. Key expansion generates a total of Nb(Nr + 1) words. The algorithm requires an initial set of Nb words, and each of the Nr rounds requires Nb words of key data. The resulting key schedule consists of a linear array of 4-byte words, denoted [wi], with i in the range $0 \le i < Nb(Nr + 1)$.

The subword()function takes a four byte input and applies the byte substitution operation and produces an output word. The rotword() takes a word [a0, a1, a2, a3] as input and performs a cyclic permutation to produce [a1, a2, a3, a0] as output word. The round constant word array rcon[i] is calculated using the below formula in finite field.

$$rcon[i] = x^{(254+i)} \bmod x^8 + x^4 + x^3 + x + 1$$

The first Nk words of the expanded key are filled with the cipher key. Every following word w[i] is equal to the xor of previous word w[i-1] and the word Nk positions earlier w[i-Nk]. For words in positions that are a multiple of Nk, a transformation is applied to w[i-1] prior to the XOR, followed by an XOR with a round constant Rcon[i]. This transformation consists of a cyclic shift of the bytes in a word rotword() and byte substitution subword(). But in key expansion of 256-bit cipher if Nk=8 and i-4 is a multiple of Nk then subword() function is applied to w[i-1] prior to the xor.

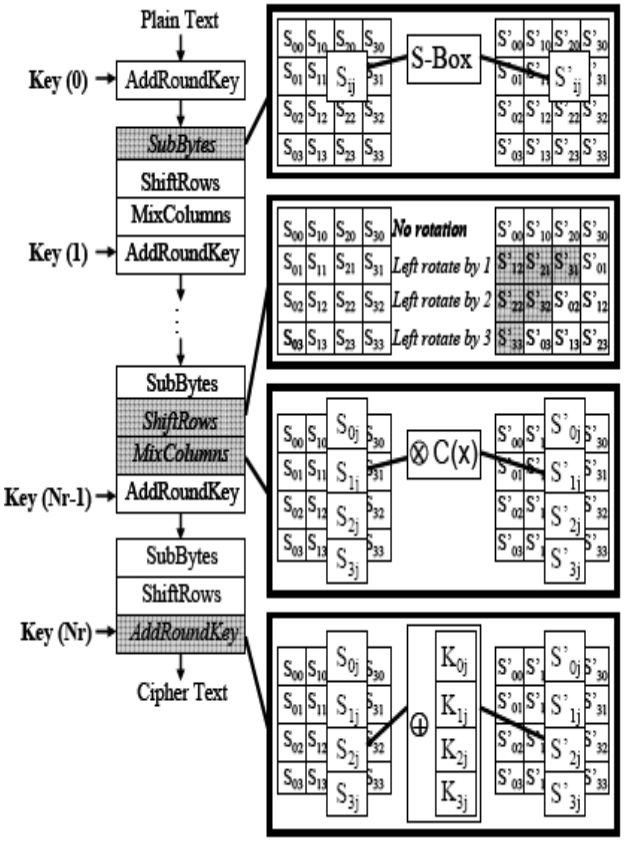

Figure 4: AES Encryption Process

B. AES Decryption Process:

The decryption of the data which was encrypted using the AES is done by inverting all the encryption operations with the same key with which it is encrypted since the AES is a symmetric encryption standard. In the decryption process the sequence of the transformations differs from that of the encryption but the key expansion for encryption and decryption are the same. However several properties of the AES algorithm allow for an equivalent decryption with the same sequence of transformations as that in encryption.

The operations of the decryption are listed below

a) Inverse Sub Bytes.
b) Inverse Shift Rows.
c) Add Round Key.
d) Inverse mix columns.

a) Inverse Sub Bytes:

This operation is same as it is in the encryption process but the only difference is the inverse of the substitution box is used here since the substitution box which we used in the encryption is invertible.

b) Inverse Shift Rows:

The inverse shift rows operation inverses the shift row operation in the encryption process by right shifting the elements in the rows.

c) Add Round Key:

The add round key process is the same as that of the one in the encryption process.

d) Inverse Mix Columns:

In inverse mix column operation the same operation in the mix column is done but with the different matrix.

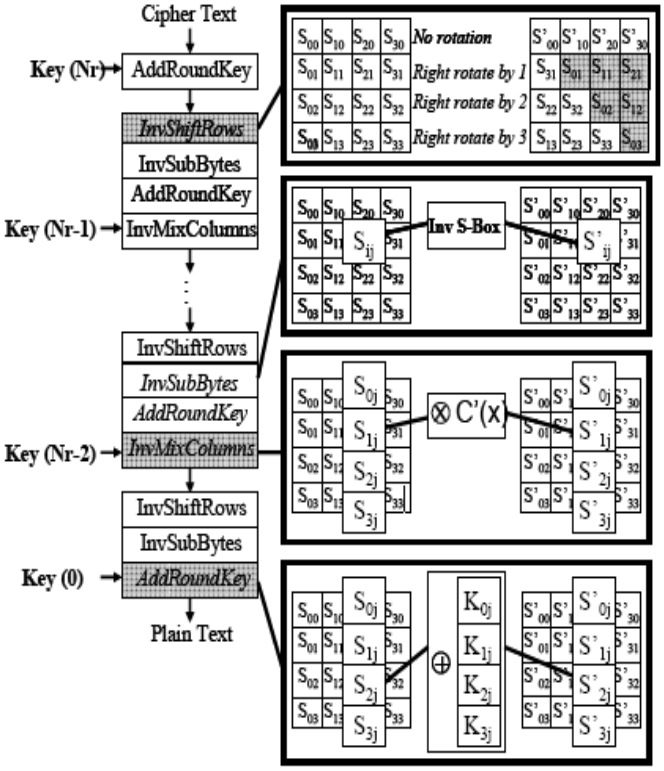

Figure 5: AES Decryption Process

III. IMPLEMENTATION

The AES algorithm is implemented using VHDL coding in Xilinx ISE 9.2. First, the algorithm is tested by encrypting and decrypting a single 128 bit block. After having an operational block cipher, the next step is to embed this block cipher in a block cipher modes of operation. Cipher feedback (CFB) shown in Figure 6 and Figure 7, is chosen since the message does not have to be padded to a multiple of the cipher block size while preventing some manipulation of the cipher text.

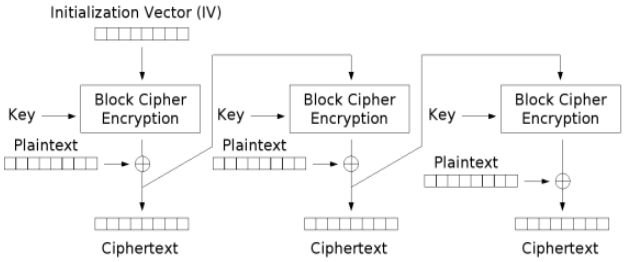

Figure 6: Encryption using Cipher Feedback (CFB).

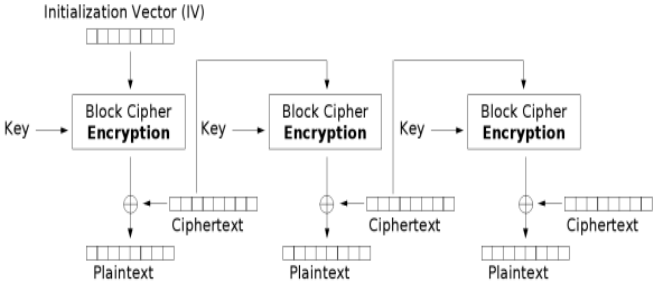

Figure 7: Decryption using Cipher Feedback (CFB).

IV. SIMULATION RESULT

A. Encryption Process (Cipher):

AES block length/Plain Text = 128bits (Nb = 4)
Key length = 128 bits (Nk = 4);
No. of Rounds = 10(Nr = 10)

Plain Text:
00112233445566778899aabbccddeeff

Key:
000102030405060708090a0b0c0d0e0f

Output/Cipher Text:
69c4e0d86a7b0430d8cdb78070b4c55a

Figure 8 represents the waveforms generated by the 128-bit complete encryption Process. The inputs are clock1 & clock2, Active High reset, 4-bit round, and 128-bit state & key as a standard logic vectors, whose output is the 128-bit cipher (encrypted) data.

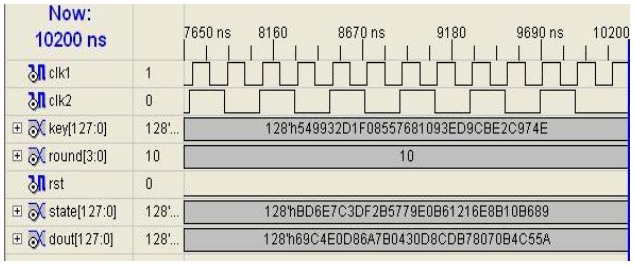

Figure 8: Simulation Waveforms of Final Round of Encryption Process

Figure 9 represents the waveforms generated by the 128-bit Key Expansion. The inputs are clock of 120ns time period, Active High reset, round, and 128-bit state as a standard logic vector, whose output is the 128-bit key for round one is generated.

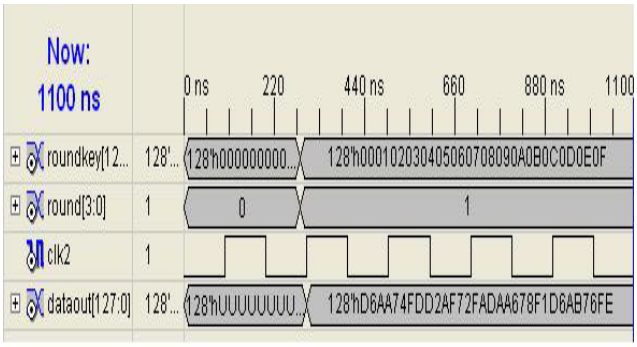

Figure 9: Simulation Waveforms of Key Expansion

B. Decryption Process (Inverse Cipher):

AES block length/Cipher Text = 128bits (Nb = 4)
Key length = 128 bits (Nk = 4);
No of Rounds = 10(Nr = 10)

Input /Cipher Text:
69c4e0d86a7b0430d8cdb78070b4c55a
Key:
000102030405060708090a0b0c0d0e0f
Output/Plain Text:
00112233445566778899aabbccddeeff

Figure 10 represents the waveforms generated by the 128-bit complete decryption Process. The inputs are clock1 & clock2, Active High reset, 4-bit round, and 128-bit state & key as standard logic vectors, whose output is the 128-bit plain text (decrypted data).

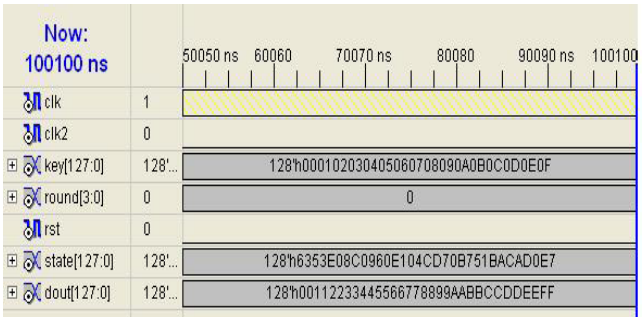

Figure 10: Simulation Waveforms of Final Round of Decryption Process

V. TESTING AND VERIFICATION

To ensure the proposed design gives better results in terms of area and throughput the design is implemented Xilinx 8.1 and FPGA device XCV600BG560-6 used for downloading. In table 2. The device utilization summary of complete algorithm i.e. AES-128, AES-192, AES-256 in same hardware is shown.

Table 2: Device utilization summary of AES Encryption

Target FPGA Device	Virtex XCV600 BG 560– 6
Optimization Goal	Speed
Maximum Operating Frequency	140.390MHz
Number of Slices	1853 out of 6912 (26%)
Number of Slice Flip Flops	512 out of 13824 (3%)
Number of 4 input LUTs	3645 out of 13824 (26%)
Number of bonded IOBs	391 out of 408 (95%)
Number of GCLKs	2 out of 4 (50%)
256x8-bit ROM	20
Encryption/Decryption Throughput	352 Mbits/sec
Total memory usage	130248 kilobyte

The parameter that compares AES candidates from the point of view of their hardware efficiency is Throughput. Encryption / Decryption Throughput = block size frequency / total clock cycles. Thus, Throughput = 128 x 140.390MHz/51 = 352 Mbits/sec.

VI. CONCLUSION

The Advanced Encryption Standard algorithm is an iterative private key symmetric block cipher that can process data blocks of 128 bits through the use of cipher keys with lengths of 128, 192, and 256 bits.

An efficient FPGA implementation of 128 bit block and 128 bit key AES cryptosystem has been presented in this paper. Optimized and Synthesizable VHDL code is developed for the implementation of both 128 bit data encryption and decryption process & description is verified using ISE 8.1 functional simulator from Xilinx. All the transformations of algorithm are simulated using an iterative design approach in order to minimize the hardware consumption. Each program is tested with some of the sample vectors provided by NIST. The throughput reaches the value of 352Mbit/sec for both encryption and decryption process with Device XCV600 of Xilinx Virtex Family.

REFERENCES

[1] Marko Mali, Franc Novak and Anton Biasizzo "Hardware Implementation of AES Algorithm" –Journal of ELECTRICAL ENGINEERING, Vol. 56, No. 9-10, 2005, 265-269.
[2] Behrouz A. Forouzan and Debdeep Mukhopadhyay "Cryptography and Network Security" (2nd edition).
[3] FIPS 197, "Advanced Encryption Standard *(AES)", November 26, 2001*.
[4] L.Thulasimani , "A Single Chip Design and Implementation of AES -128/192/256 Encryption Algorithms"- International Journal of Engineering Science and Technology, Vol. 2(5), 2010, 1052-1059.
[5] Nation Institute of Standards and Technology (NIST), Data Encryption Standard (DES), National Technical Information Service, Sprinfgield, VA 22161, Oct. 1999.
[6] J. Daemen and V. Rijmen, "AES Proposal: Rijndael", AES Algorithm Submission, September 3, 1999.
[7] J. Nechvatal et. al., Report on the development of Advanced Encryption Standard, NIST publication, Oct 2, 2000.

[8] FIPS 197, "Advanced Encryption Standard (AES)", November 26, 2001.
[9] K. Gaj and P. Chodowiec, Comparison of the hardware performance of the AES candidates using reconfigurable hardware, in The Third AES Candidates Conference, printed by the National Institute of Standards and Technology.
[10] L.Thulasimani , "Design And Implementation of Reconfigurable Rijndael Encryption Algorithms For Reconfigurable Mobile Terminals"- International Journal on Computer Science and Engineering, Vol. 02, No. 04, 2010, 1003-1011.