FPGA Implementation of High Speed AES Algorithm for Improving The System Computing Speed

Vijaya Kumar. B.1[#1], T. Thammi Reddy.2[#2]

#1. Dept of Electronics and Communication, G.P.R.Engineering College, Kurnool, JNTU Antapur, AP, India. Phone no.9492592534,

#2. Associate professor, Dept of Electronics and Communication, G.P.R. Engineering College, Kurnool, JNTU Antapur, AP, India. Phone no.9618085446,

Abstract: An implementation of high speed AES algorithm based on FPGA is presented in this paper in order to improve the safety of data in transmission. The mathematical principle, encryption process and logic structure of AES algorithm are introduced .so as to reach the purpose of improving the system computing speed, the pipelining and parallel processing methods were used. 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 Encryption 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.

1.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 algorithm.

The algorithm is composed of three main parts: Cipher, Inverse Cipher and Key Expansion. Cipher converts data to an unintelligible form called cipher text 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) the AES algorithm, the number of rounds to be performed during the execution of the algorithm is dependent on the key length. Table.1

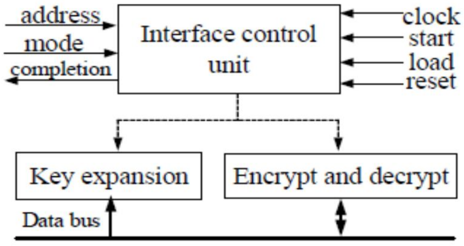
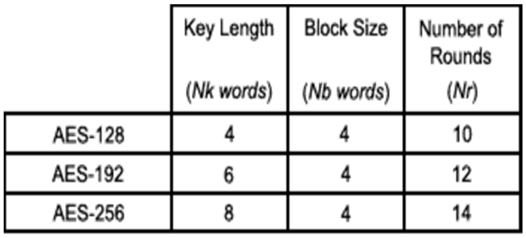


Figure 1. AES module Architecture



	Key Length (Nk words)	Block Size (Nb words)	Number of Rounds (Nr)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Key-Block-Round Combinations

AES operates on a 4x4 array of bytes (referred to as "state"). The algorithm consists of performing four different simple operations.

THESE OPERATIONS ARE:

- **SUB BYTES**
- **SHIFT ROWS**
- **MIX COLUMNS**
- **ADD ROUND KEY**

Sub bytes perform byte substitution which is derived from a Multiplicative inverse of a finite field.

Shift rows shifts elements from a given row by an offset Equal to the row number.

Mix columns step transforms each column using an Invertible linear transformation.

Add round Key step takes a 4x4 block from a expanded key (Derived from the key), and XORs it with the "state". AES is composed of four high-level steps. These are:

1. KEY EXPANSION
2. INITIAL ROUND
3. ROUNDS
4. FINAL ROUND

The Key Expansion step is performed using key schedule. The Initial Round consists only of an Add Round Key operation. The Rounds step consists of a Sub Bytes, Shift Rows, Mix Columns, and an Add Round Key operation. The number of rounds in the Rounds step varies from 10 to 14 depending on the key size. Finally, the Final Round performs a Sub Bytes, Shift Rows, and an add Round key operations

Decryption in AES is done by performing the inverse operations of the simple operations in reverse order. However, as shown later on in this paper, because of the block cipher mode of operation used, decryption is Implemented but never used.

II. THE AES ALGORITHM

The AES encryption and decryption processes for a 128-bit plain text block are shown in Fig. 2 and 3. The AES Algorithm specifies three encryption modes: 128-bit, 192-bit, and 256-bit. Each cipher mode has a corresponding number of rounds Nr based on key length of Nk words .The state block size, termed Nb, is constant for all encryption modes. This 128-bit block is termed the state. Each state is comprised of 4 words. A word is subsequently defined as 4 bytes. Table1 Shows the possible key/state block/round combinations.

A. ENCRYPTION PROCESS:

The Encryption and decryption process consists of a number of different transformations applied consecutively over the data block bits, in a fixed number of iterations, called rounds. The number of rounds depends on the length of the key used for the encryption process. For key length of 128 bits, the number of iteration required are10. (Nr = 10). As shown in Fig. 2, each of the first Nr-1 rounds consists of 4 transformations: Sub Bytes (), Shift Rows (), Mix Columns () & Add Round Key ().

The four different transformations are described in detail below:

1). SUB BYTES TRANSFORMATION (SB):

It is a non-linear substitution of bytes that operates independently on each byte of the State using a substitution table (S box). This S-box which is Invertible is constructed by first taking the multiplicative Inverse in the finite field GF (28) with irreducible Polynomial m(x) = x8 + x4+ x3 + x + 1. The element {00} is mapped to it. Then affine transformation is applied (over GF (2)).

2). SHIFT ROWS TRANSFORMATION (SR):

Cyclically shifts the rows of the State over different offsets. The operation is almost the same in the decryption process except for the fact that the shifting offsets have different values.

3) MIX COLUMNS TRANSFORMATION (MC):

This transformation operates on the State column-by-column, treating each column as a four-term polynomial. The columns are Considered as polynomials over GF (28) and multiplied by modulo x4 + 1 with a fixed polynomial a(x) = {03} x3+ {01} x2+ {01} x+ {02}.

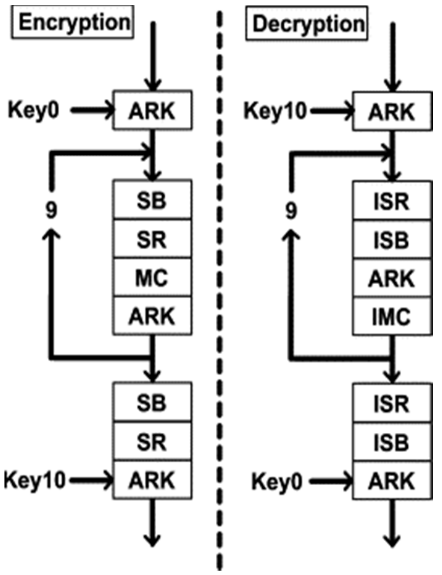


Fig.2 Block diagram representation of AES algorithm (Decryption and Encryption)

4) ADD ROUND KEY TRANSFORMATION (ARK):

In this transformation, a Round Key is added to the State by a simple bitwise XOR operation. Each Round Key consists of Nb words from the key expansion.

Those Nb words are each added into the columns of the State. Key Addition is the same for the decryption process. Figure. AES Encryption and Decryption Process

Key Expansion: Each round key is a 4-word (128-bit) array generated as a product of the previous round key, a constant that changes each round, and a series of S-Box lookups for each 32-bit word of the key The key schedule Expansion generates a total of Nb (Nr+1) words.

B. DECRYPTION PROCESS

For decryption, the same process occurs simply in reverse order – taking the 128-bit block of cipher text and converting it to plaintext by the application of the inverse of the four operations. Add Round Key is the same for both encryption and decryption. However the three other functions have inverses used in the decryption process: Inverse Sub Bytes, Inverse Shift Rows, and Inverse Mix Columns.

This process is direct inverse of the Encryption process. All the transformations applied in Encryption process are inversely applied to this process. Hence the last round values of both the data and key are first round inputs for the Decryption process and follows in decreasing order.

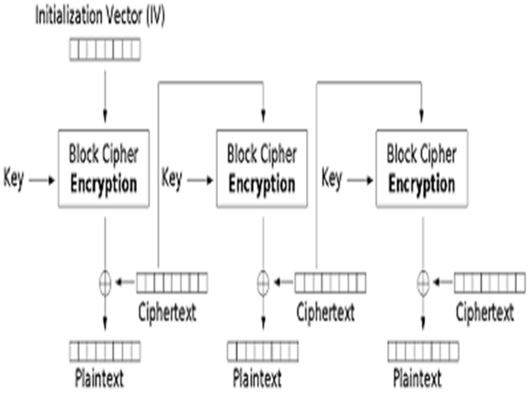


Figure 4: Encryption using Cipher Feedback (CFB

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 4 and Figure 5, 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.

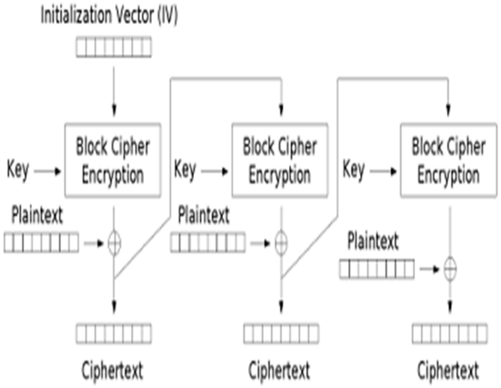


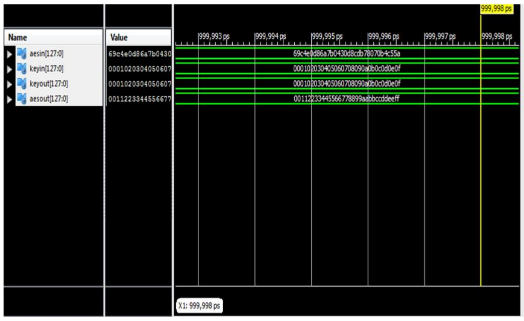
Figure 5: Decryption using Cipher Feedback (CFB)

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 6 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 standard logic vectors, whose output is the 128-bit cipher (encrypted)



```
Name             Value               999,993 ps  999,994 ps  999,995 ps  999,996 ps  999,997 ps  999,998 ps
aesin[127:0]     69c4e0d86a7b0430    69c4e0d86a7b0430d8cdb78070b4c55a
keyin[127:0]     0001020304050607    000102030405060708090a0b0c0d0e0f
keyout[127:0]    0001020304050607    000102030405060708090a0b0c0d0e0f
aesout[127:0]    0011223344556677    00112233445566778899aabbccddeeff
                                                                                                999,998 ps
X1: 999,998 ps
```

Figure6: Simulation Waveforms of final round of Encryption process

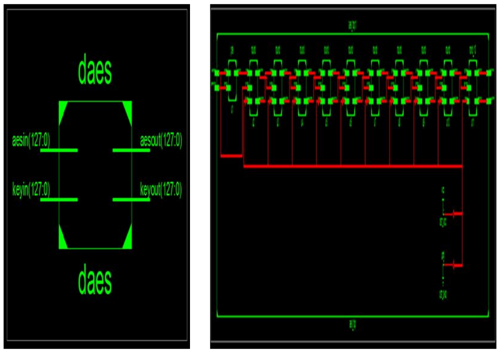
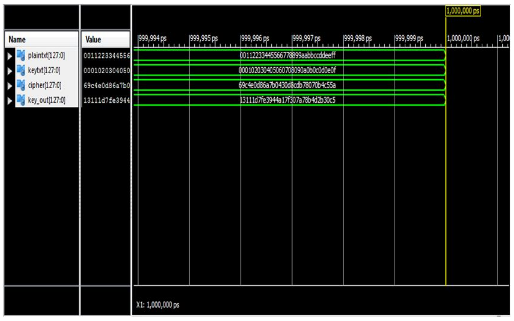


Fig: RTL Schematic of Encryption Process

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



```
Name             Value             999,994 ps  999,995 ps  999,996 ps  999,997 ps  999,998 ps  999,999 ps  1,000,000 ps  1,00
plaintxt[127:0]  0011223344556     00112233445566778899aabbccddeeff
keytxt[127:0]    0001020304050     000102030405060708090a0b0c0d0e0f
cipher[127:0]    69c4e0d86a7b0     69c4e0d86a7b0430d8cdb78070b4c55a
key_out[127:0]   13111d7fe3944     13111d7fe3944a17f307a78b4d2b30c5
                                                                                                1,000,000 ps
X1: 1,000,000 ps
```

Figure 7: Simulation Waveforms of final round of Decryption process

Figure 7 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).

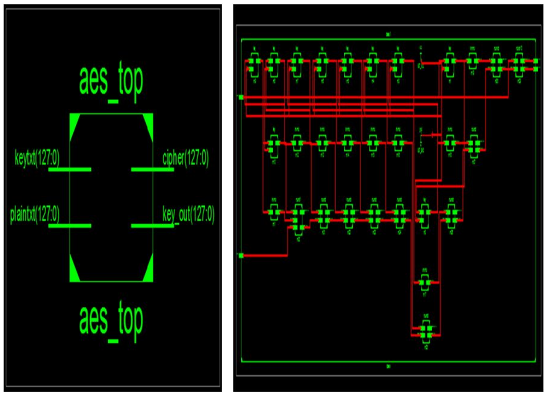


Fig: RTL Schematic of Decryption Process

V. TESTING AND VERIFICATION

The synthesis & mapping results of AES design are summarized in Table 2

Target FPGA device	Vertex XCV600BG 560-6
Optimization goal	Speed
Maximum Operating Freq	140.390 MHz
No. Of slices	1853 out of 6912(26%)
No. Of Slice flip flops	512 out of 13824(3%)
No. Of 4-i/p LUTs	3645 out of 13824(26%)
No. Of bonded IOBs	391 out of 408 (95%)
No. Of GCLK’	2 out of 4(50%)
256x8-bit ROM	20
Encryption/Decryption Throughput	352 Mbps
Total memory uses	130248 Kbytes

Table 2: Results of FPGA Implementation of AES

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 Vertex 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] Beerhouse A. Forouzan and Debden Mukhopadhyay "Cryptography

[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, Springfield, VA 22161, Oct. 1999.

[6] J. Daemen and V. Rijmen, "AES Proposal: Rijndael", AES

[7] J Algorithm Submission, September 3, 1999. 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.