VIRTEX-7 IMPLEMENTATION OF AES 128 BIT CIPHER

Kunal Lala[1] , Swati Singh[2] , Saumya Sharma[3]

[1]PG, Department of Electronics and Communication Engineering, KIET, Ghaziabad (India)

[2,3]UG, Department of Electronics and Communication Engineering,

Raj Kumar Goel Institute of Technology for Women, Ghaziabad (India)

ABSTRACT

This paper presents our experience in implementing the Advance Encryption Standard algorithm using 128 bit block of data and 128 bit cipher key. We have simulated our design for all the standard test vectors provided by [1]. We have used VHDL for coding and Xilinx 13.3 tool for simulation and synthesis. We have provided a best possible tradeoff between the area i.e. maximum logic utilization and throughput i.e. speeds of transmission. Our implementation on Virtex -7 keeping the target device XC7VX415T-3-FFG1157 provides a throughput of 46.03 Gbits/sec at a frequency of 359.74MHz using a maximum logic utilization of 67 percent in the form of Bounded I/0 blocks.

Keywords: AES, Cryptography, Encryption.

I INTRODUCTION

Cryptography plays an important role in the security of data. It enables us to store sensitive information or to transmit it across insecure networks so that unauthorized persons cannot read it. The urgency of secure exchange of digital data resulted in large quantities of different encryption algorithm (with public key algorithms) and symmetric encryption algorithms (with private key algorithms). Electronically ,Symmetric key algorithms are in general are much faster to execute than asymmetric key algorithms. AES originated from the initiative of the National Institute of Standards and Technology (NIST) in 1997 to select a new symmetric key encryption algorithm. From the initial candidates due to combination of security, performance, efficiency, ease of implementation and flexibility,Rijndael algorithm was selected as the Advanced Encryption Standard(AES).

The algorithm is composed of three main parts: cipher, inverse cipher and key expansion. Cipher converts data into an unintelligible form called cipher text while Inverse cipher converts data back into original form called plain text. 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. The number of rounds to be performed during the execution of of the algorithm depends on the key length.[2]

II. ALGORITHM SPECIFICATION

The length of the input block, the output block and the State is 128 bits. This is represented by *Nb* = 4, which reflects the number of 32-bit words (number of columns) in the State. The AES algorithm uses a round function for both its Cipher and Inverse Cipher that is composed of four different byte-oriented transformations:

1) A substitution table (S-box) is used for byte substitution.

2) Different offset is used for shifting rows of the State array..

3) Within each column of the State array is mixed data.

4) A Round Key is added to the State.

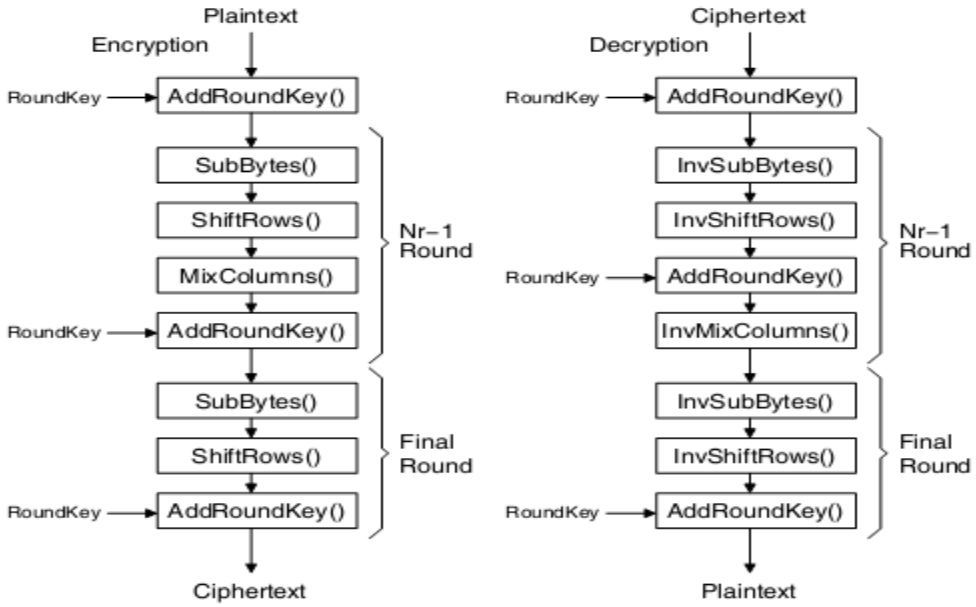

Fig. 1 Encryption and Decryption Process[2]

2.1 The Sub Bytes() 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 is invertible. S- box or the Sub bytes transformation mainly consists of two sub steps. Multiplicative inverse of each and every byte over the irreducible polynomial. ($x^8 + x^4 + x^3 + x + 1$). Then apply affine transformation.

$$\begin{bmatrix} b_0' \\ b_1' \\ b_2' \\ b_3' \\ b_4' \\ b_5' \\ b_6' \\ b_7' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

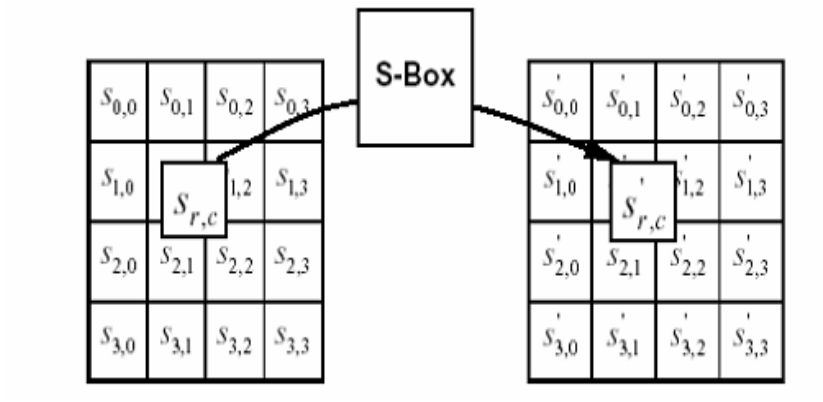

Fig2. Sub bytes Transformation[1]

2.2 Shift Rows Transformation: In the ShiftRows() transformation , the last three rows of the bytes of the State are cyclically shifted over different numbers of bytes (offsets). The first row, $r = 0$, is not shifted the row [1] is shifted by one, row [2] is shifted by two and row [3] by three. This is shown in Fig. 3.

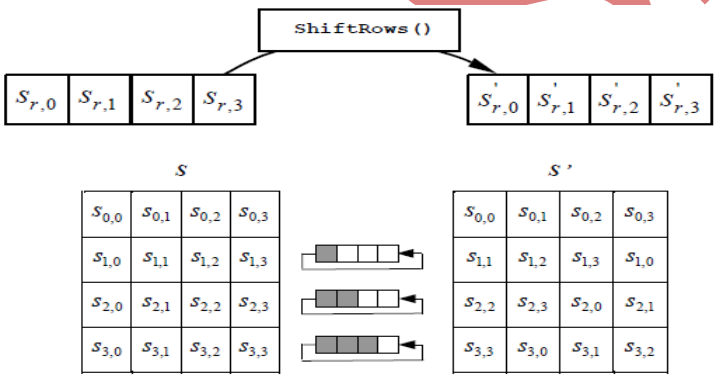

Fig 3. Shift Rows Transformation [1]

2.3 Mix columns ( ):-The Mix Columns treating each column as a four-term polynomial operates on the State column-by-column. This operates on four bytes or a word at a time unlike other operations which operate on a byte. Here each column is multiplied by a polynomial. The expansion of this matrix is entirely based on the multiplication of polynomial coefficient in G.F (2^8) [4]. The columns are considered as polynomials over GF (2^8) and multiplied modulo $x^4 + 1$ with a fixed polynomial $a(x)$, given by the following equation

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

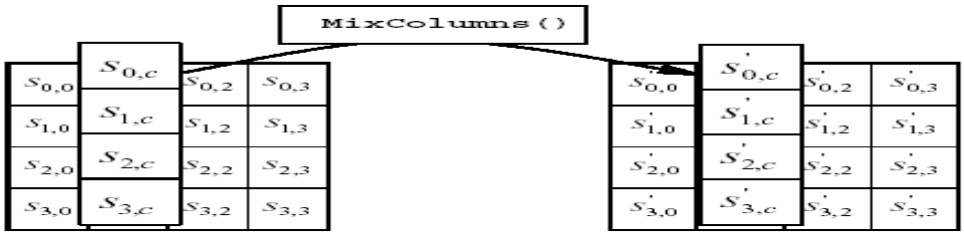

Fig 4. Mix Columns Transformation[1]

2.4 Add Round Key Transformation In the AddRoundKey() transformation, a simple bitwise XOR operation is used to add the round key to the state. From key expansion, each Round Key consists of Nb words. Then Nb words are each added into the columns of the State. Key Addition is the same for the decryption process.

III. ALGORITHM IMPLEMENTATION

We have developed a synthesizable VHDL code for AES encryption and decryption. The coding has been done in VHDL. Xilinx 13.3 tool has been used for synthesis and simulation. The results have also been optimized for throughput and area. After several synthesis on different Virtex 7 families we found the best suitable results on XC7VX415T3FFG1157.

IV RESULTS

4.1 Synthesis Result

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	5232	515200	1%
Number of slice LUTs	4294	257600	1%
Number of fully used LUT-FF pairs	3840	5686	67%
Number of bonded IOBs	386	600	64%
Number of Block RAM/FIFO	49	880	5%
Number of BUFG/BUFGCTRLs	1	32	3%

Table 1.Device utilization summary of XC7VX415T-3-FFG1157

Timing summary

Speed grade:-2

Minimum period: 2.780ns (Maximum frequency:359.738 MHz)

Minimum input arrival time before clock: 2.022ns

Maximum output required time after clock: 0.51ns

Throughput=46.043 Gbits/sec

4.2 Simulated Result

4.2.1 Encryption: - The simulated waveform of test bench for encryption has input and cipher key as "ab23c57d8f43e377988cac3131df489e" and "24356fd38a9efcd354da31d28afe4ac" respectively. The encrypted data is received as "acbbb1abcae0a7cf03705e633360cb17". Shown in fig: 5

4.2.2 Decryption: - The simulated waveform of test bench for decryption has input and cipher key as "acbbb1abcae0a7cf03705e633360cb17" and " 24356fd38aefcd354da31d28af11e4ac" respectively . The decryption results in the original data as " ab23c57d8f43e377988cac3131df489e". Shown in fig: 6

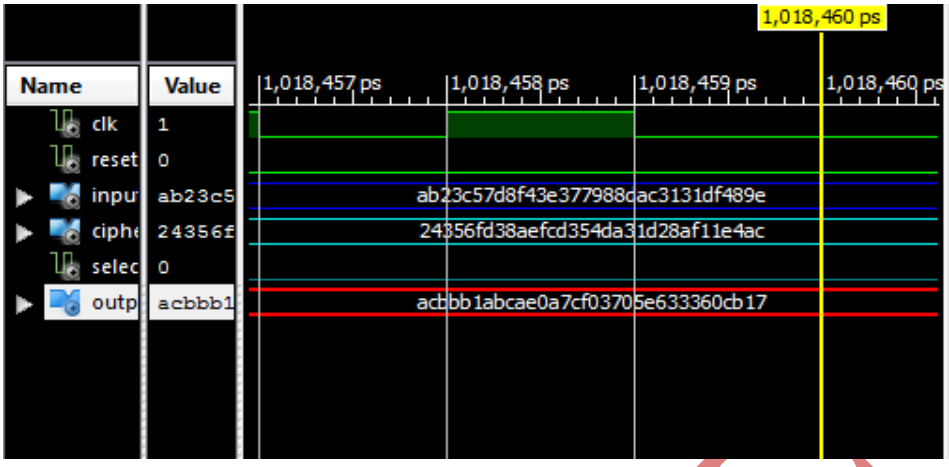

Fig 5. Simulation waveform of test bench for Encryption [4]

Fig 6. Simulation waveform of test bench for Decryption[4]

V CONCLUSION AND FUTURE SCOPE

In future the modules for Encryption and Decryption can be combined , the code can also be extended for ASCII coding.

REFERENCES

[1]Kunal lala ,Ajay kumar , Amit kumar , "Design and synthesis of fully pipelined AES architecture" ICCE 2012.

[2] J. Daemen and V. Rijmen, "AES Proposal: Rijndael", AES Algorithm Submission, September 3, 1999 .

[3] Rajendra Manteena "A VHDL Implemetation of the Advanced Encryption Standard-Rijndael Algorithm" March 23,2004.

[4] FIPS 197, "Advanced Encryption Standard (AES)", November 26, 2001.