 ISSN: 2321-8169
2326 – 2329

Encryption and Decryption Using Rijndael Algorithm

Manisha Mankar
Department of Electronics Engineering
Priyadarshini College of Engineering
Nagpur, Maharashtra,India
manshree379@gmail.com

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

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

Abstract—Rijndael algorithm is an efficient cryptographic technique consist of different operations in iterative looping approach in order to minimize hardware consideration, with block size of 128 bit, lookup table implementation of S-box. It includes generation of ciphers for encryption and inverse ciphers for decryption by performing four rounds of transformations. This paper presents 192 bit key size cipher. Synthesizing and implementation of the VHDL code is carried out on Xilinx-Project Navigator ISE 14.5 software.

Keywords-*Rijndael Algorithm,Key Expansion,Encryption,Decryption,Cryptography.*

*****

I. INTRODUCTION

Cryptography is the science of information security which has become very critical in modern computing system to secure data transmission and storage. The need for privacy has become a major priority as widespread use of personal communication devices. The exchange of digital data in cryptography results in different algorithm classified into two cryptographic mechanism: symmetric key in which same key is used for encryption and decryption which are fast and easier to implement than asymmetric key algorithm.

Rijndael algorithm to be introduced in October 2000 replacing the DES algorithm. Rijndael is a symmetric byte oriented iterated block cipher that can process 128 bits using keys with length of 128,192,256 bits.

II. RIJNDAEL ALGORITHM

Rijndael algorithm composed of three main parts: Cipher, Inverse Cipher and Key expansion. Cipher converts data in a coded form called Cipher text and inverse cipher converts data back into its original form called plaintext. Key expansion generates a key schedule used in cipher and inverse cipher procedure and composed of specific number of rounds. Number of rounds is dependent on the key length. Rijndael algorithm specifies three encryption: 128 bit, 192 bit, 256 bit. Number of rounds Nr based on key length of Nk words. Nb is constant for all versions.

Table 1: Rijndeal Key/Block/Round Size

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

In cipher and inverse cipher, round function is composed of four different byte oriented transformations-Sub Byte, Shift Rows , mix column and Add round key .

Encryption and Decryption process for 192 bit key shown in figure 1.

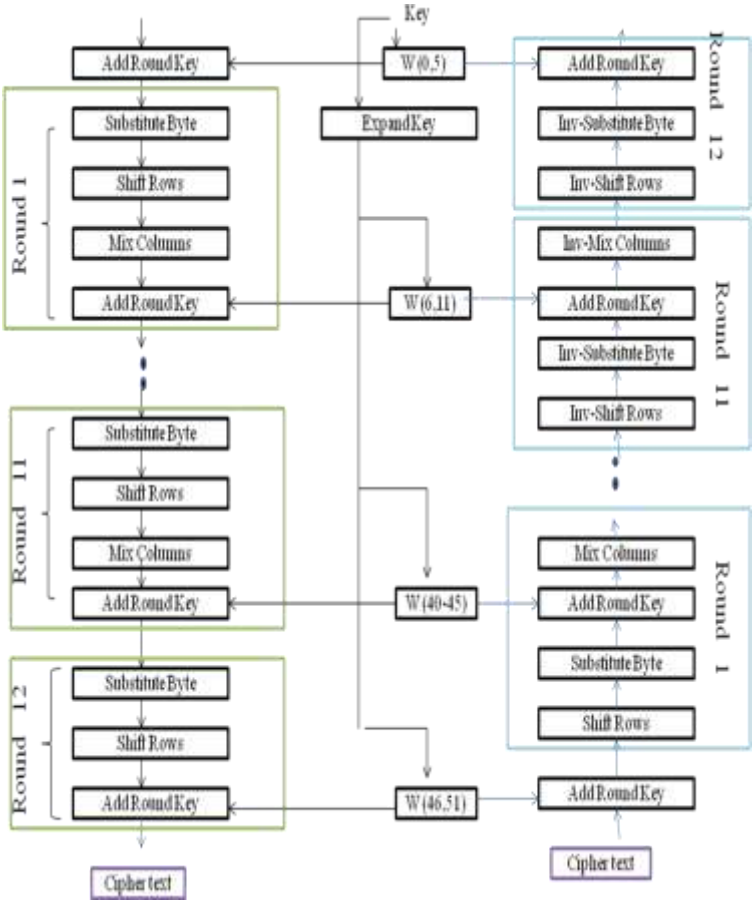


Figure 1:Rijndael Algorithm

ISSN: 2321-8169
2326 – 2329

The data / plaintext will first XOR with initial key and basic round operation repeated Nr-1 times.Nr depends on the key length. Last round will execute only three functions.

A. *Subbyte and Inverse Subbyte*

In this, each input byte of the state matrix is independently replaced by another byte from look-up table called S-box is replaced with its multiplicative inverse in GF (2^8) with the element {00}being mapped onto itself ,followed by affine transformation over GF(2^8).For decryption ,inverse S-box is obtained by applying inverse affine transformation followed by multiplicative inversion in GF(2^8) shown in figure 2.

$S'_{r,c} \leftarrow$ S-box ($S_{r,c}$)

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

Sbox ⟹

$S'_{0,0}$	$S'_{0,1}$	$S'_{0,2}$	$S'_{0,3}$
$S'_{1,0}$	$S'_{1,1}$	$S'_{1,2}$	$S'_{1,3}$
$S'_{2,0}$	$S'_{2,1}$	$S'_{2,2}$	$S'_{2,3}$
$S'_{3,0}$	$S'_{3,1}$	$S'_{3,2}$	$S'_{3,3}$

Figure 2: Subbyte Transformation

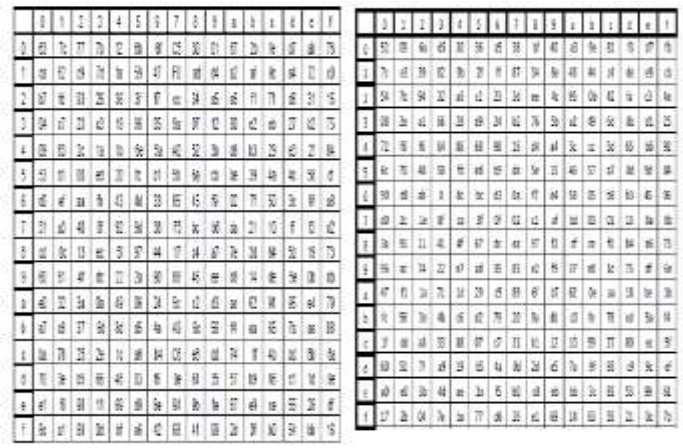


Figure 3: Rijndael S-box and inverse S-box

B. *Shift Row and Inverse Shift Row*

It is a cyclic shift operation, each row is repeated cyclically to the to the left using 0,1,2,3byte offset for encryption as shown in figure 4,while for decryption to the right.

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

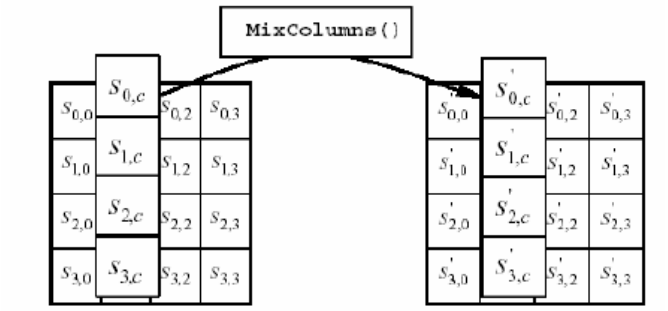
⟹

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,1}$	$S_{1,2}$	$S_{1,3}$	$S_{1,0}$
$S_{2,2}$	$S_{2,3}$	$S_{2,0}$	$S_{2,1}$
$S_{3,3}$	$S_{3,0}$	$S_{3,1}$	$S_{3,2}$

Figure 4: Rijndael Shift Row Operation

C. *Mix Column and Inverse Mix Column*

In this transformation, each column of the state matrix is multiplied by a constraint fix matrix is shown in figure 5,for i = 0,1 2, 3 respectively.



$S'_{0,c}$ = ({02}.$S_{0,c}$) Ex-or ({03}.$S_{1,c}$) Ex-or $S_{2,c}$ Ex-or $S_{3,c}$
$S'_{1,c}$ = $S_{0,c}$ Ex-or ({02}.$S_{1,c}$) Ex-or({03}.$S_{2,c}$) Ex-or $S_{3,c}$
$S'_{2,c}$ = $S_{0,c}$ Ex-or $S_{1,c}$ Ex-or ({02}.$S_{2,c}$) Ex-or({03}. $S_{3,c}$)
$S'_{3,c}$ = ({03}.$S_{0,c}$) Ex-or $S_{1,c}$ Ex-or $S_{2,c}$ Ex-or ({02}. $S_{3,c}$)

Similarly for decryption we compute inverse mix column by multiplying each column of state matrix by constant fix matrix.

$S'_{0,c}$ = ({0e}.$S_{0,c}$) Ex-or ({0b}.$S_{1,c}$) Ex-or ({0d}. $S_{2,c}$) Ex-or ({09}. $S_{3,c}$)
$S'_{1,c}$ = ({09}.$S_{0,c}$) Ex-or ({0e}.$S_{1,c}$) Ex-or ({0b}. $S_{2,c}$) Ex-or ({0d}. $S_{3,c}$)
$S'_{2,c}$ = ({0d}.$S_{0,c}$) Ex-or ({09}.$S_{1,c}$) Ex-or ({0e}. $S_{2,c}$) Ex-or ({0b}. $S_{3,c}$)
$S'_{3,c}$ = ({0b}.$S_{0,c}$) Ex-or ({0d}.$S_{1,c}$) Ex-or ({09}. $S_{2,c}$) Ex-or ({0e}. $S_{3,c}$)

⟹ Mix Columns ⟹

$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

Figure 5: Rijndael Mix-column

D. *Add Round Key*

The output of mix column is XOR-ed with corresponding Round Sub key derived from user key. The Add round key step is same for encryption and decryption.

in_0	in_4	in_8	in_{12}
in_1	in_5	in_9	in_{13}
in_2	in_6	in_{10}	in_{14}
in_3	in_7	in_{11}	in_{15}

Input Byte

ISSN: 2321-8169
2326 – 2329

Round keys

$K_{0,0}$	$K_{0,1}$	$K_{0,2}$	$K_{0,3}$	$K_{0,4}$	K_{05}
$K_{1,0}$	$K_{1,1}$	$K_{1,2}$	$K_{1,3}$	$K_{1,4}$	$K_{1,5}$
$K_{2,0}$	$K_{2,1}$	$K_{2,2}$	$K_{2,3}$	$K_{2,4}$	$K_{2,5}$
$K_{3,0}$	$K_{3,1}$	$K_{3,2}$	$K_{3,3}$	$K_{3,4}$	$K_{3,5}$

Output Bytes

out_0	out_4	out_8	out_{12}
out_1	out_5	out_9	out_{13}
$0ut_2$	out_6	out_{10}	out_{14}
out_3	out_7	out_{11}	out_{15}

Figure 6: Rijndael Add Round key

E. *key expansion*

The AES key expansion algorithm takes as input a 6-word key and produces a linear array of 52 words. Each word contains 32 bytes which means each subkey is 192 bits long.

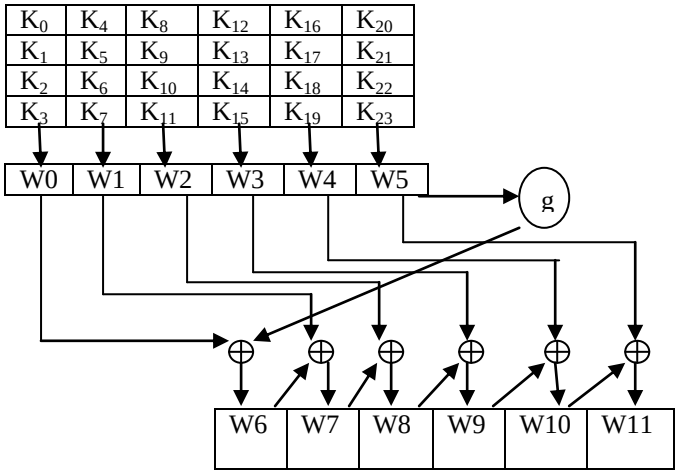


Figure 7: Rijndael key Expansion

III. RIJNDAEL IMPL.EMENTATION

Rijndael algorithm is implemented using VHDL coding in Xilinx ISE 14.5 .Algorithm is tested by block size. 128 bits. only one block data is encrypted at a time, the number of clock cycle necessary to encrypt a single block of data is equal to number of cipher rounds. Cipher feedback is used in encryption and decryption process.

IV. SIMULATION RESULTS

A) Encryption Process
Rijndael block length/Plaintext = 128 bits (Nb=4)
Key length = 192 bits (Nk = 6)
No. of rounds = 12 (Nr = 12)
Plaintext:
00112233445566778899aabbccddeeff
Key:
000102030405060708090a0b0c0d0e0f1011121314151617
Cipher text:
dda97ca4864cdfe06eaf70a0ec0d7191

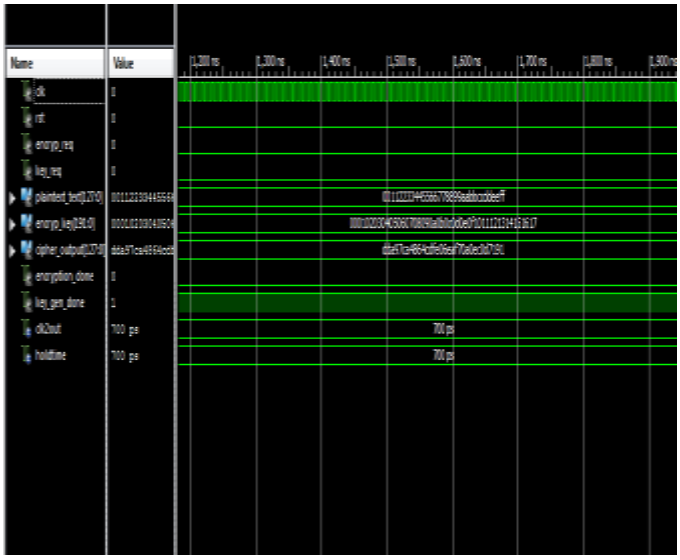


Figure 8: waveforms of Encryption process

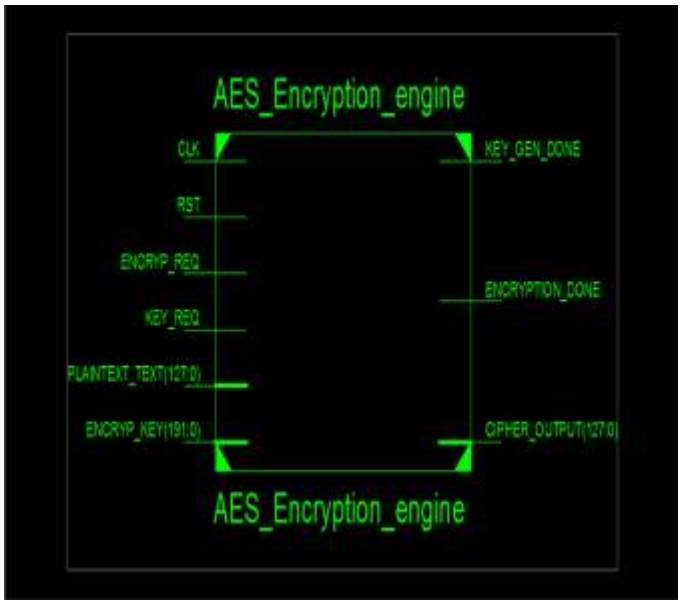


Figure 9: RTL Schematic of Encryption process

B) Decryption Process
Rijndael block length/Plaintext = 128 bits (Nb=4)
Key length = 192 bits (Nk = 6) No. of rounds = 12 (Nr = 12)
Input/Cipher text:
dda97ca4864cdfe06eaf70a0ec0d7191
Key:
000102030405060708090a0b0c0d0e0f1011121314151617
Plaintext:
00112233445566778899aabbccddeeff

Figure 10: waveforms of Decryption process

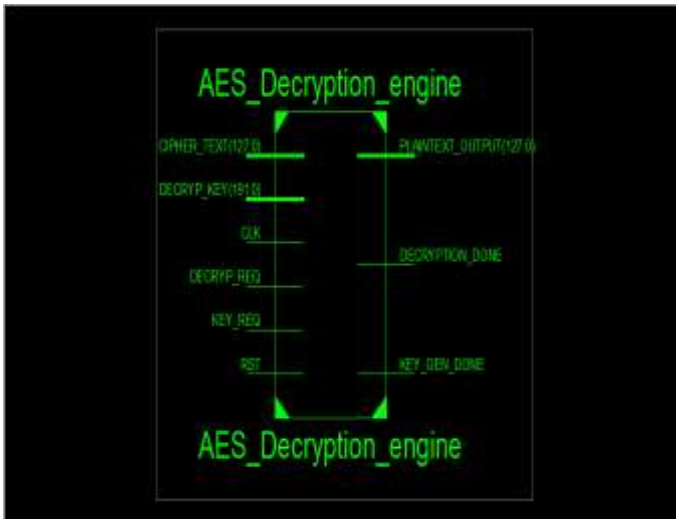


Figure 11: RTL Schematic of Decryption process

V. CONCLUSION

Rijndael Algorithm is an iterative symmetric block cipher that can process data blocks of 128 bits and key lengths of 128,192,256 bits. In this paper, 128 bit block and 192 bit key is used.VHDL code is developed and results are verified using Xilinx ISE 14.5 Simulator both for encryption and decryption process.

REFERENCES

[1] Bin Liu, Student Member, IEEE, and Bevan M. Baas, Senior Member, IEEE; "Parallel AES Encryption Engines for Many-Core Processor Arrays", IEEE TRANSACTIONS ON COMPUTERS, VOL. 62, NO. 3, MARCH 2013

[2] Kamalika Datta ,Vishal Shrivastav ,Indranil Sengupta, Hafizur Rahaman "Reversible Logic Implementation of AES Algorithm"8th international conference on design & technology Integrated systems in nanoscale era,978-1-4673-6040-1/13/$31.00,2013 IEEE.

[3] N.Anitha Christy and P.Karthigaikumar "FPGA implementation of AES algorithm using Composite Field Arithmetic"International conference on devices circuits and systems,978-1-4577-1546-4/12/$26.00,2012 IEEE

[4] Hong Trang;Nguyen Van Loi;"An efficient FPGA implementation of the advanced Encryption Standard algorithm"978-1-4673-0309-5/12,2012,IEEE.

[5] Nabihah Ahmad, N.; Hasan, R.; Jubadi, W.M; "Design of AES S-Box using combinational logic optimization", IEEE Symposium on Industrial Electronics & Applications (ISIEA), pp. 696-699, 31.00,2010 IEEE.

[6] Marcelo Barcelos,Ricardo Reis "An IP of an Advanced Encryption Standard For Altera Devices",IEEE Symposium on Integrated circuits and systems Design (SBCCI'02)0-7695-1807-9/02,2002, IEEE.

[7] Hrushikesh S.Deshpand, Kailash J.Karande, Altaaf O.Mulani "Efficient Implementation Of AES Algorithm on FPGA" International Conference on Communication and Signal Processing,April 3_5,2014,IEEE.

[8] Girish Kumar P,Mahesh Kumar "Implementation of AES algorithm using verilog"International Journal of VLSI and Embedded Systems,vol-4,Article 05090;june 2013.

[9] Hrushikesh S.Deshpand, Kailash J.Karande, Altaaf O.Mulani "Efficient Implementation Of AES Algorithm on FPGA" Progress In Science in Engineering Reaserch Journal ,PISER 11,vol.02,ISSN 2347-6680 (E),2014.

[10] Vijaya Kumar.B, T.Thammi Reddy "Fpga Implementation of High speed AES Algorithm For Improving The System Computing speed" International Jouranal of Computer trends and Technology(IJCTT)-VOLUME 4 Issue 9-Sep 2013.

[11] J. Nechvatal et.al., Report on the development of Advanced Encryption Standard, NIST Publication, Oct,2000.

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

[13] National Institute of Standards and Technology (NIST), Data Encryption Standard (DES), National Technical Information Service, Spring field, VA 22161, Oct. 1999.

[14] N Radhika, Obili Ramesh, Priyadarshini, "Design and Verification of Area-Optimized AES Based on FPGA using verilog HDL"International Journal of Engineering Trends and Technology-volume 4 Issue9-Sep 2013.