IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2320-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 10, Issue 3, March 2021

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 7.512

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003201

An Efficient Implementation of AES Algorithm Using VHDL

S.Angeline Priyadharsini(A.P)

Department of EEE, Sriram Engineering College, Perumalpattu, Tamilnadu, India

ABSTRACT: We present an efficient implementation of AES (Advanced Encryption Standard) algorithm using VHDL. The Symmetric, or secret key algorithms, a cryptography method for both encryption and decryption calculations the same key value is used are becoming more popular. Secret key cryptography uses conventional algorithm that is Advanced Encryption Standard (AES) algorithm. This contribution investigates the AES encryption and decryption cryptosystem with regard to FPGA and Very High Speed Integrated Circuit Hardware Description language (VHDL).Optimized and Synthesizable VHDL code is developed for the implementation of both 128-bit data encryption and decryption process. ModelSim software is used for simulation. Some results are verified using test bench for the input plaint text of 128 bit and output Cipher Text of 128 bit.

KEYWORDS: Advanced Encryption Standard (AES), Block cipher, plain text, Cipher text, Encryption, Decryption, composite field arithmetic, Key Expansion, VHDL, Field Programmable Gate Array (FPGA)

I. INTRODUCTION

The **Advanced Encryption Standard** (**AES**) is a specification for the encryption of electronic data introduced and used by the U.S.National Institute of Standards and Technology (NIST) in 2001[1].

AES is based on the Rijndael cipher developed by two Belgian cryptographers, Joan Daemen and Vincent Rijmen, who gave a proposal to NIST during the AES selection process. Rijndael is a group of ciphers with different key and block sizes.

For AES, NIST selected 3 members of the Rijndael family, each with a block size of 128 bits, but three different key lengths: 128, 192 and 256 bits.

AES has been adopted by the U.S. government and it is used worldwide now. It overcomes the Data Encryption Standard (DES), which was published in 1977. The algorithm described by AES is a symmetric-key algorithm, meaning the identical key is used for both encrypting and decrypting the data.

In the United States, AES was announced by the NIST as U.S. FIPS PUB 197 (FIPS 197) on November 2001. This announcement continued a five-year standardization process in which fifteen competing designs were presented and evaluated, before the Rijndael cipher has been selected as the most suitable.

AES became effective as a federal government standard on May 26, 2002 after approved by the Secretary of Commerce. AES is included in the ISO 18033-3 standard. AES is available in many standards of encryption, and it is the Firstly accessible in public and open cipher approved by the National Security Agency (NSA) for top secret information when used in an NSA approved cryptographic module[1].The name *Rijndael* is derived from names of the two inventors (Joan Daemen and Vincent Rijmen).The structure of this paper is as follows: Section I introduces about AES algorithm used in cryptography. In Section II, provides a description of the four transformations required for the AES algorithm. Section III discusses result and implementation of AES algorithm. Section IV describes conclusion of the paper.

II. DESCRIPTION OF AES ALGORITHM

The AES algorithm is a symmetric block cipher that can encrypt and decrypt information. Encryption converts the data to an unreadable form called cipher-text. The decryption process converts the data back into its original form, which is called plain-text. Symmetric Key is the one in which the sender and the receiver uses a same key for data encryption and decryption of data is called as symmetric key. Symmetric- key encryption can use either stream ciphers or block ciphers. Stream ciphers encrypt the digits (typically bytes) of a message one at a cycle. Block ciphers take a number of bits and encrypt them as a single unit, padding of the plaintext so that it is a multiple of the block size.

The original data is converted into another format by using encryption algorithm of the cryptography. In cryptography, cipher text (or cypher text) is the result of encryption performed on plaintext using an algorithm, called a cipher. Cipher

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003201

text is also known as encrypted or encoded information because it contains a form of the original plaintext that is unreadable by a human or computer without the proper cipher to decrypt it. To an outsider it simply looks like a jumble or assortment of meaningless data.

In cryptography, plaintext is information a sender wishes to transmit to a receiver. Plaintext is used as input to an encryption algorithm; the output is usually termed cipher text particularly when the algorithm is a cipher. Code text is less often used, and almost always only when the algorithm involved is actually a code. In some systems, however, multiple rounds of encryption are used, in such case the output of one encryption algorithm becomes input for the next plaintext.
The master secret key is used to encrypt the multiple data and master issues the public and private key of the system. It is the keyword which runs the cipher generating algorithm.

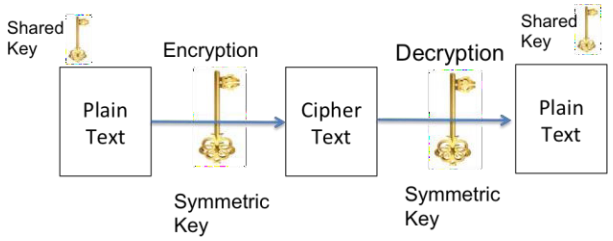

Fig 1 Encryption/Decryption Processes

A. AES encryption

AES is based on a design principle known as a substitution- permutation network, combination of both substitution and permutation, and it is fast in both software and hardware. Unlike its previous algorithm of DES, AES does not use a Feistel network, AES of Rijndael which has a fixed block size of 128 bits, and a key size of 128, 192, or 256 bits.
AES operates on a 4×4 column-major order matrix of bytes, termed the *state*, although some versions of Rijndael have a larger block size and have additional columns in the state[7].

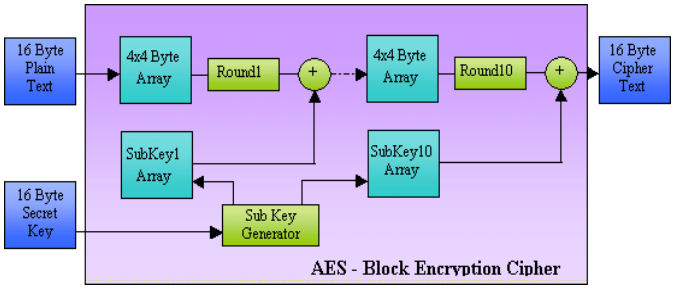

Fig 2 Block diagram of Encryption

The key size used for an AES cipher specifies the number of repetitions of transformation rounds that convert the input, called the plaintext, into the final output, called the cipher text. The number of cycles of repetition is as follows:

	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

Table 1 cycle of AES

Each round consists of several processing steps, each containing four similar but different stages, including one that depends on the encryption key itself. A set of reverse rounds are applied to transform cipher text back into the original plaintext using the same encryption key[14].

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003201

B. Rijndael algorithm

Rijndael is a substitution linear transformation cipher, not requiring a Feistel network. It uses triple discreet invertible uniform transformations (layers). Specifically, these are: Linear Mix Transform; Non-linear Transform and Key Addition Transform. Even before the first round, a simple key addition layer is performed, which adds to security. Thereafter, there are Nr-1 rounds and then the final round. The transformations form a State when started but before completion of the entire process.

The State can be thought of as an array, structured with 4 rows and the column number being the block length divided by bit length (for example, divided by 32). The cipher key similarly is an array with 4 rows, but the key length divided by 32 to give the number of columns. The blocks can be interpreted as one-dimensional arrays of 4-byte vectors.

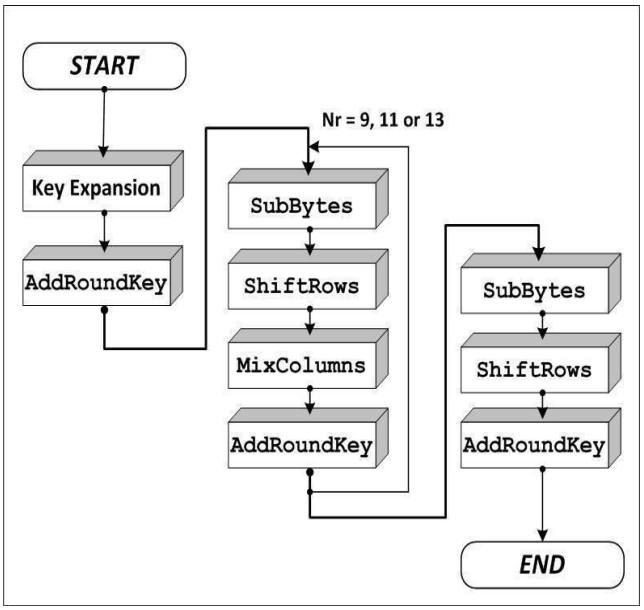

Fig 3 Block diagram of Rijndael algorithm

The exact transformations occur as follows: the byte sub transformation is nonlinear and operates on each of the State bytes independently - the invertible. S-box (substitution table) is made up of 2 transformations. The shift row transformation sees the State shifted over variable offsets. The shift offset values are dependent on the block length of the State. The mix column transformation sees the State columns take on polynomial characteristics over a Galois Field values (2^8), multiplied $x^4 + 1$ (modulo) with a fixed polynomial. Finally, the round key transform is XORed to the State. The key schedule helps the cipher key determine the round keys through key expansion and round selection[8].

1. Key Expansions—round keys are derived from the cipher key using Rijndael's key schedule. AES requires a separate 128-bit round key block for each round plus one more.
2. To avoid attacks based on simple algebraic properties, the S-box is constructed by combining the inverse function with an invertible affine transformation. The S-box is also chosen to avoid any fixed points (and so is a derangement),
3. Initial Round

i.e.,also any opposite fixed points,

- AddRoundKey—each byte of the state is combined with a block of the round key using bitwise xor.

4. Rounds

- SubBytes—a non-linear substitution step where each byte is replaced with another according to a lookup table.
- ShiftRows—a transposition step where the last three rows of the state are shifted cyclically a certain number of steps.
- MixColumns—a mixing operation which operates on the columns of the state, combining the four bytes in each column.
- AddRoundKey

5. Final Round (no MixColumns)

- SubBytes
- ShiftRows

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003201

- AddRoundKey

C. RIJNDAEL 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. All the transformations applied in encryption process are inversely applied to this (Nr + 1) words. The decryption process is direct inverse of the encryption process[2][12].

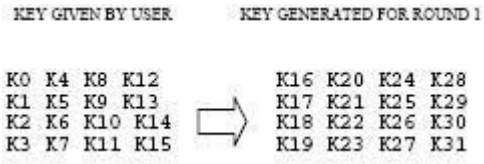

Fig 4 Key Expansion

SUB BYTES TRANSFORMATION

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 (2^8) with irreducible polynomial $m(x) = x^8 + x^4 + x^3 + x + 1$. The element {00} is mapped to itself. Then affine transformation is applied (over GF (2^8)).

In the Sub Bytes step, each byte in the state is replaced with its entry in a fixed 8-bit lookup table, S; $b_{ij} = S(a_{ij})$. The Sub Bytes step, each byte $a_{i,j}$ in the *state* matrix is replaced with a Sub Byte $S(a_{i,j})$ using an 8-bit substitution box, the Rijndael S-box. This operation provides the non-linearity in the cipher. The S-box used is derived from the in multiplicative inverse over GF (2^8), known to have good non-linearity properties.

i.e., .While performing the decryption, Inverse Sub Byte step is used, this requires first taking the affine transformation and then finding the multiplicative inverse (just reversing the steps used in Sub Bytes step)[10].

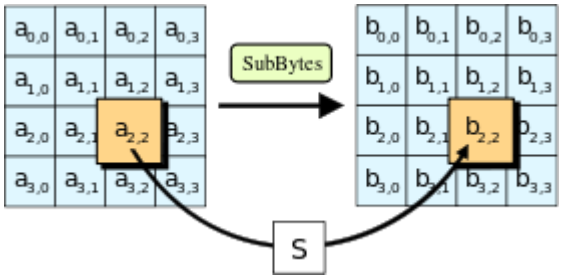

Fig 5: Sub byte transformation block

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Fig 6 S-BOX

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003201

SHIFT ROWS TRANSFORMATION

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. In the Shift Rows step, bytes in each row of the state are shifted cyclically to the left.

The Shift Rows step operates on the rows of the state; it cyclically shifts the bytes in each row by a certain offset. For AES, the first row is left unchanged. Each byte of the second row is shifted one to the left. Similarly, the third and fourth rows are shifted by offsets of two and three respectively. For blocks of sizes 128 bits and 192 bits, the shifting pattern is the same. Row n is shifted left circular by n-1 bytes. In this way, each column of the output state of the Shift Rows step is composed of bytes from each column of the input state. (Rijndael variants with a larger block size have slightly different offsets).

For a 256-bit block, the first row is unchanged and the shifting for the second, third and fourth row is 1 byte, 3 bytes and 4 bytes respectively- this change only applies for the Rijndael cipher when used with a 256-bit block, as AES does not use 256-bit blocks. The importance of this step is to avoid the columns being linearly independent, in which case, AES degenerates into four independent block ciphers[10].

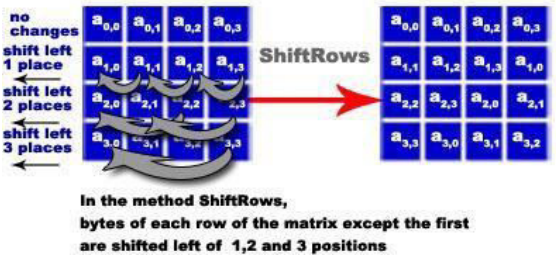

Fig 7: Shift rows transformation block

MIX COLUMNS TRANSFORMATION

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 (2^8) and multiplied by modulo $x^4 + 1$ with a fixed polynomial a(x) = {03} x^3+ {01} x^2+ {02} x. In the Mix Columns step, each column of the state is multiplied with a fixed polynomial *c(x)*.

In the Mix Columns step, the four bytes of each column of the state are combined using an invertible linear transformation. The Mix Columns function takes four bytes as input and outputs four bytes, where each input byte affects all four output bytes. Together with Shift Rows, Mix Columns provides diffusion in the cipher.

During this operation, each column is multiplied by a fixed matrix:

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

The Mix Columns step can also be viewed as a multiplication by the shown particular MDS matrix in the finite fieldGF (2^8). This process is described further in the article Rijndael mix columns[11].

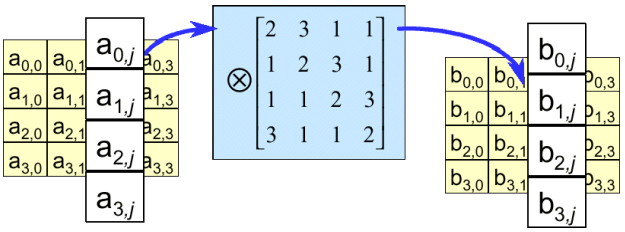

Fig 8: Mix columns transformation block

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003201

ADD ROUND KEY TRANSFORMATION

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.
In the Add Round Key step, each byte of the state is combined with a byte of the round sub key using the XOR operation (⊕).
In the Add Round Key step, the sub key is combined with
the state. For each round, a sub key is derived from the main key using Rijndael's key schedule; each sub key is the same size as the state. The sub key is added by combining each byte of the state with the corresponding byte of the sub key using bitwise XOR.
XOR state is done with 128-bits of the round key. It is again processed by column (though effectively a series of byte operations). It is designed to be as simple as possible[13].

Matrix multiplication is composed of multiplication and addition of the entries, and here the multiplication operation

DECRYPTION

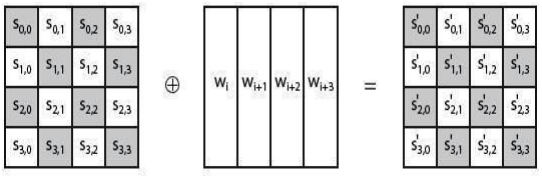

Fig 9: Add round key transformation

can be defined as this: multiplication by 1 means no change, multiplication by 2 means shifting to the left, and multiplication by 3 means shifting to the left and then performing XOR with the initial unshifted value. After shifting, a conditional XOR with 0x1B should be performed if the shifted value is larger than 0xFF. (These are special cases of the usual multiplication in GF (2^8).) Addition is simply XOR.
In more general sense, each column is treated as a polynomial over GF (2^8) and is then multiplied modulo x^4+1 with a fixed polynomial $c(x) = 0x03 \cdot x^3 + x^2 + x + 0x02$. The coefficients are displayed in their hexadecimal equivalent of the binary representation of bit polynomials from GF (2) [x].
Decryption is a reverse of encryption which inverse round transformations to computes out the original plaintext of an encrypted cipher-text in reverse order. The round transformation of decryption uses the functions AddRoundKey, InvMixColumns, InvShiftRows, and InvSubBytes successively[4].

III. IMPLEMENTATION AND RESULT OF AES ALGORITHM

The AES algorithm is implemented using Verilog hardware descriptive language and simulated using a ModelSim ALTERA WEB EDITION 6.3g_p1. The algorithm is tested by encrypting and decrypting a single 128 bit block.
Encryption simulation was successfully completed by the use of key expansion and transformations of shift Rows, sub bytes, mix columns, add round keys without pipelining stages shown in fig 10.

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003201

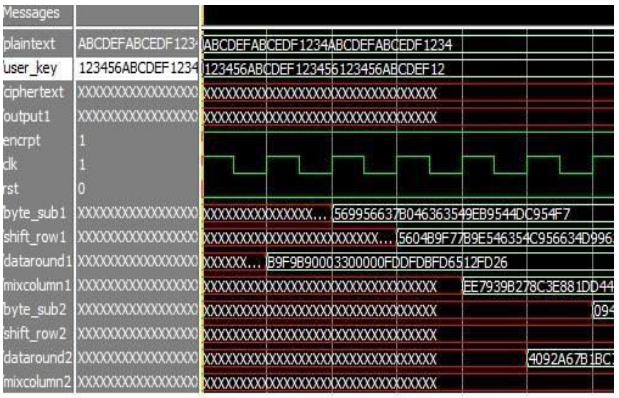

Fig 10 Encryption Result

Decryption simulation was successfully completed by the use of key expansion and transformations of inverse shift Rows, inverse sub bytes, inverse mix columns, inverse add round keys shown in fig 11.

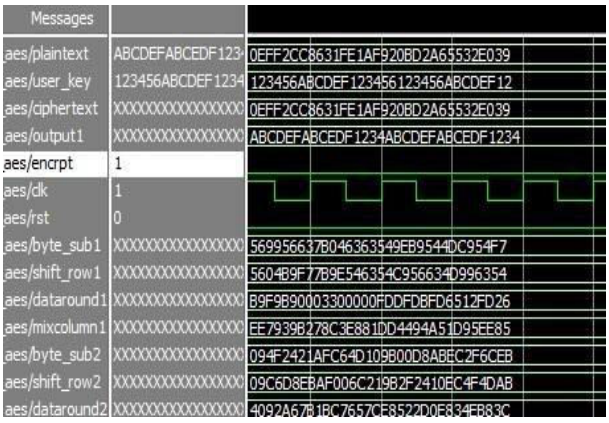

Fig 11 Decryption Result

IV. CONCLUSIONS

The design and strength of all key lengths of the AES algorithm (i.e., 128, 192 and 256) are sufficient to protect classified information up to the secret level. Top secret information will require use of either the 192 or 256 key lengths. The implementation of AES in products intended to protect national security systems and/or information must be reviewed and certified by NSA prior to their acquisition and use. The data block of 128 bit and 128 bit key has been implemented and verified using model simulator. 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.

ACKNOWLEDGMENT

First and foremost, I place this project work on the feet of GOD ALMIGHTY who is the power of strength in each step of progress towards the successful completion of project.

I take this opportunity to thank my parents who gave full support of my education, our Management of Sriram Educational Trust, my Project Supervisor Mrs.S.Anjeline priyadharsini, M.E, Assistant Professor, and Project Coordinator Mr.K.Vasudevan M.E, Assistant Professor who have been a constant source of inspiration and guidance throughout my project. And I enunciate my deep sense of gratitude and whole hearted thanks to Department of Electrical and Electronics for her exquisite and valuable advice throughout the project.

| e-ISSN: 2319-8753, p-ISSN: 2320-6710| www.ijirset.com | Impact Factor: 7.512|

|| Volume 10, Issue 3, March 2021 ||

DOI:10.15680/IJIRSET.2021.1003201

REFERENCES

[1] Shelke R.B, Mr. Patil A.P, Dr (Mrs.). Patil S.B, "VLSI Based Implementation of Single Round AES Algorithm", *IOSR Journal of Electronics and Communication Engineering (IOSR-JECE).*

[2] Bin Liu, Student Member, IEEE, and Bevan M. Baas, "Parallel AES Encryption Engines for Many-Core Processor Arrays", *Senior Member, IEEE* volume. 62, number 3, March 2013.

[3] Debasis Das and Abhishek Ray, "Implementation of Low Power RC5 Algorithm in Xilinx FPGA", *Journal of computer science and engineering*, volume 1, issue 1, May 2010.

[4] Hoang Trang, University of Technology and Nguyen Van Loi, *IC Design Research & Education Center (ICDREC)*, VietNam National University HoChiMinh City "An Efficient FPGA Implementation of the Advanced Encryption Standard Algorithm".

[5] Minal Moharir and Dr A V Suresh, "A Novel Approach Using Advanced Encryption Standard To Implement Hard Disk Security", *International Journal of Network Security & Its Applications (IJNSA)*, Vol.4, No.1, January 2012.

[6] Hamdan.O.Alanazi, B.B.Zaidan, A.A.Zaidan, Hamid A.Jalab, M.Shabbir and Y. Al-Nabhani, "New Comparative Study Between DES, 3DES And AES Within Nine Factors" *Journal of Computing*, Volume 2, Issue 3, MARCH 2010.

[7] Xinmiao Zhang, Student Member, IEEE and Keshab K. Parhi, Fellow, IEEE, "High-Speed VLSI Architectures For The AES Algorithm", *IEEE Transactions on Very Large Scale Integration (VLSI) systems,* Vol. 12, No. 9, September 2004.

[8] Amandeep Kaur, Puneet Bhardwaj and Naveen Kumar, "FPGA Implementation Of Efficient Hardware For The Advanced Encryption Standard" *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, Volume-2, Issue-3, February 2013.

[9] Ibtihal Mohamed Abdullateef Fadul and Tariq Mohamed Hassan Ahmed, "Enhanced Security of Rijndael Algorithm Using Two Secret Keys", *International Journal of Security and Its Applications* Volume. 7, No. 4, July, 2013.

[10] Suja Chackochan, K.Mathan, "Low Power and Area Optimized VHDL Implementation of AES", *International Journal of Science and Research (IJSR)* Volume 3 Issue 3, March 2014.

[11] M.S.Arun, V.Saminathan, "Parallel AES Encryption with Modified Mix-Columns for Many Core Processor Arrays", *International Journal of Engineering Science and Innovative Technology (IJESIT)* Volume 3, Issue 3, May 2014.

[12] Dalal Naeem Hmood, "A Random Key Generation Approach for Rijndael Algorithm", *Journal of Al-Nahrain University* Vol.15 (3), September, 2012.

[13] Muhammad Nazrul Islam, Md. Monir Hossain Mia, Md. Foizul Islam,
M.A. Matin, "An Efficient Approach for Increasing Security to Symmetric Data Encryption", ***International*** *Journal of Computer Science and Network Security*, VOL.8 No.4, April 2008.

[14] Renjith V Ravi, Dr.R.Mahalakshmi, "Simulation and Error Analysis of Rijndael Algorithm", *International Journal of Advanced Research in Computer Science and Software Engineering*, Volume 4, Issue 5, May 2014

INNO SPACE
SJIF Scientific Journal Impact Factor

Impact Factor:
7.512

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर
NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details