Implementation of AES Algorithm Using FPGA

Dhairya Senghani (21BEC111)
Institute of Technology, Nirma University,
Ahmedabad, Gujarat

Aditya Vartak (21BEC132)
Institute of Technology, Nirma University,
Ahmedabad, Gujarat

Abstract— **The Advanced Encryption Standard (AES) is a widely adopted symmetric-key encryption algorithm that plays a crucial role in securing data in various applications. This paper provides a concise introduction to the AES-128 algorithm, including its history, key features, and its significance in contemporary information security.**

Keywords— Field Programmable Gate Array (FPGA) AES, Round Key Operation, Mixed Column Operation.

I. Introduction

The majority of communication in the modern world takes place via electronic media. Data security is essential to this kind of communication. A growing amount of data handling is required in computer networks and communication technologies in order to handle the massive amounts of data and information that public communication networks must share. Thus, cryptography is always growing, making sensitive data increasingly susceptible to automated eavesdropping and offering high data transfer efficiency and security. Therefore, it is imperative to safeguard data against malevolent attacks.

The Advanced Encryption Standard (AES) is a symmetric-key encryption algorithm, meaning the same key is used for both encryption and decryption. It was established to replace the aging Data Encryption Standard (DES) and is characterized by its speed, security, and flexibility.

AES offers varying key lengths, with AES-128 being one of the most commonly used versions. AES looks to withstand all known assaults well and is based on strong, well-publicized mathematical foundations. Since it has been out there for a while and has been the focus of extensive examination by researchers worldwide, there isn't actually a known flaw or backdoor. AES is a successful method of protecting enormous volumes of information and economic value. The National Security Agency (NSA) has approved it, making it the first publicly accessible open cypher.

Using a pipeline pattern, the AES algorithm highlights its throughput in light of the current state of domestic and international research. The largest benefit is an increase in system throughput; yet, there is a glaring drawback: on-chip resource consumption. Additionally, a low-throughput terminal now requires the use of the AES algorithm, therefore a high-safety, low-cost reduced AES system with a reduced hardware structure is created and proven on the Altera. This system's benefits include a smaller chip area, increased cost-effectiveness, fast speed, and high dependability. These will successfully encourage the usage of the AES algorithm in terminal devices.

A. Historical Context:

The need for a more secure encryption standard became apparent as computational capabilities increased. The National Institute of Standards and Technology (NIST) initiated an open competition in 1997 to select a replacement for DES. AES emerged as the winner in 2001, with its 128-bit key length variant being chosen as the standard.

II. Related Work

A. ***Sonali A. Varhade, N. N. Kasat-*** The Advanced Encryption Standard (AES) has emerged as a widely accepted and secure symmetric encryption technique, offering high computational efficiency and versatility in various applications. This literature review delves into the extensive research conducted on optimizing AES implementations, particularly in the context of hardware architectures and algorithms. The review highlights key advancements, such as the development of optimized S-box and Mix Columns operations on dedicated ASIC and FPGA platforms, which enhance the security and speed of encryption/decryption. The potential for expanding key size and input block to increase security and throughput is also explored. With a focus on FPGA-based implementations, this survey identifies the evolving landscape of AES encryption techniques and underscores the promising opportunities for future work in multimedia communications and the support of multiple key lengths and modes of operation.

B. *A.Amaar, I. Ashour and M. Shiple* - presents a compact implementation of advanced encryption standard AES using different devices of FPGA technology There are various ways to accomplish this trade-off between speed and area. The suggested architecture uses a 128-bit data route for the plaintext and the cypher key. The suggested design implements a 128-bit data-path for the plaintext and the cypher key. The suggested approach aims to reduce the number of unnecessary latches and shift registers in between the main block that consumes the area "SBOX" in order to conserve space. The routing rails are twisted in order to apply the rejected shift raw block. Combination gates implement the mix column. ModelSim is used to simulate the suggested minimum area AES architecture that is provided by VHDL in order to confirm its usefulness as a preliminary verification tool. Additionally, Xilinx is used to synthesise and apply the suggested algorithm (translate, fit, place, and route).

C. *Pritamkumar N. Khose and Prof. Vrushali G. Raut*- AES was standardized by National Institute of Standards and Technology in 2001 became Federal Information Processing Standard FIPS-197. Where Rijndael algorithm by Joan Daeman and VicentRijimen was selected as standard AES algorithm . The AES is private or symmetric block cipher which uses the same key for encryption and decryption is more suitable for faster implementation. The AES is a symmetric key for both encryption and decryption. AES cryptography algorithm is capable of encrypting and decrypting block size 128 bit data using cipher keys of 128, 196 or 256 bits (AES128, AES196 and AES256) .The proposed design has ability to defend against fault and glitch attacks with small increase area than conventional design. Proposed S-box is capable to reduce hardware resources and defend against glitch attacks.

D. *Atul M. Borkar, Dr. R. V. Kshirsagar, and Mrs. M. V. Vyawahare* - Advanced Encryption Standard was presented by Atul M. Borkar, Dr. R. V. Kshirsagar, and Mrs. M. V. Vyawahare. It can be constructed using pure hardware or implemented in software. But a speedier, more adaptable solution is provided by Field Programmable Gate Arrays (FPGAs). In relation to FPGA and the Very High Speed Integrated Circuit Hardware Description language (VHDL), this study examines the AES algorithm. VHDL code that can be synthesised is optimised and simulated using software. In order to reduce the amount of hardware used, an iterative design method is used to mimic all of the encryption and decryption transformations .

E. *Adnan Mohsin Abdulazeez and Ari Shawkat Tahir*-Two architectures—one for the 128-bit AES Encryption process and the other for the 128-bit AES Decryption process—have been proposed in this work. Both architectures rely on an iterative structure and modifications, such as lookup tables for decryption, optimization of each clock cycle to incorporate maximum number of operations to improve the throughput and reduce hardware resources, merging transformation (SubByte and ShiftRow in the encryption process, and Inverse SubByte and Inverse ShiftRow in the decryption process).

III. Key Features of AES-128:

A. *128-Bit Key Length:* AES-128 employs a 128-bit key, which provides a large number of possible keys, making it resistant to brute force attacks.

B. *Block Cipher*: AES-128 encrypts data in fixed-size blocks (128 bits). It operates on data in 16-byte blocks.

C. *Substitution-Permutation Network:* AES-128 utilizes a series of substitution and permutation operations to ensure data diffusion and confusion.

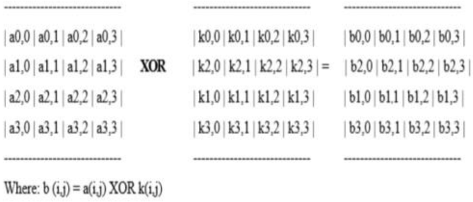
D. *Rounds*: It consists of 10 rounds for key expansion and data transformation, ensuring high security.

E. *Efficiency:* AES-128 is known for its efficient encryption and decryption operations, making it suitable for various applications.

IV. Working of AES

A. *Important Operations: Before moving on to the working, a few operations are needed to be understood.*

1) *Add Round Key operation:* We XOR data word (State Matrix) and key word, bitwise. (Figure 1)

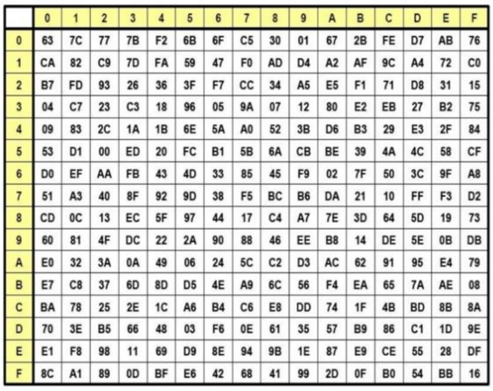


```
---------------------------          ---------------------------     ---------------------------
| a0,0 | a0,1 | a0,2 | a0,3 |         | k0,0 | k0,1 | k0,2 | k0,3 |     | b0,0 | b0,1 | b0,2 | b0,3 |
| a1,0 | a1,1 | a1,2 | a1,3 |   XOR   | k2,0 | k2,1 | k2,2 | k2,3 | =   | b2,0 | b2,1 | b2,2 | b2,3 |
| a2,0 | a2,1 | a2,2 | a2,3 |         | k1,0 | k1,1 | k1,2 | k1,3 |     | b1,0 | b1,1 | b1,2 | b1,3 |
| a3,0 | a3,1 | a3,2 | a3,3 |         | k3,0 | k3,1 | k3,2 | k3,3 |     | b3,0 | b3,1 | b3,2 | b3,3 |
---------------------------          ---------------------------     ---------------------------

Where: b (i,j) = a(i,j) XOR k(i,j)
```

Figure 1: Add Round Key

2) *Subbyte Operation*: We replace the 128 bit input as per the S-box which is given in Figure 2. For example, if the input is 12, it will be replaced by C9.



	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

Figure 2: S-Box

3) *ShiftRows Operation:* The 128 bit input is divided in 16 parts and arranged in a 4x4 matrix. The first row is kept as it is, the next row is shifted left once (rotated), the next row twice and the last row is shifted 3 times. (figure 3)

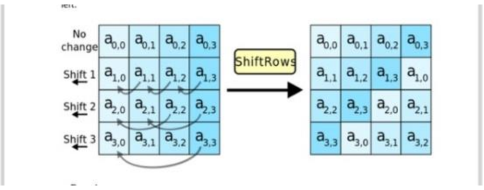
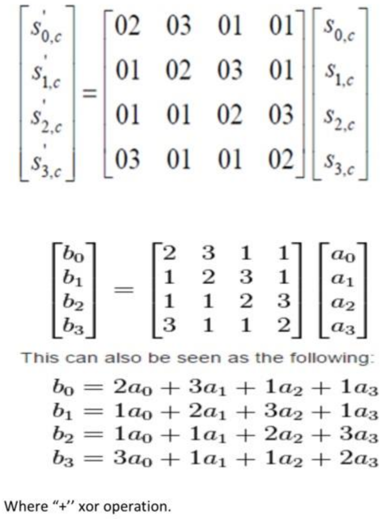

No change
Shift 1
Shift 2
Shift 3
ShiftRows
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,1}$ $a_{1,2}$ $a_{1,3}$ $a_{1,0}$
$a_{2,2}$ $a_{2,3}$ $a_{2,0}$ $a_{2,1}$
$a_{3,3}$ $a_{3,0}$ $a_{3,1}$ $a_{3,2}$

Figure 3: ShiftRows Operation

4) *MixedColumn Operation*: Each column from the State Matrix is multiplied with a fixed matrix. (Figure 4)



$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

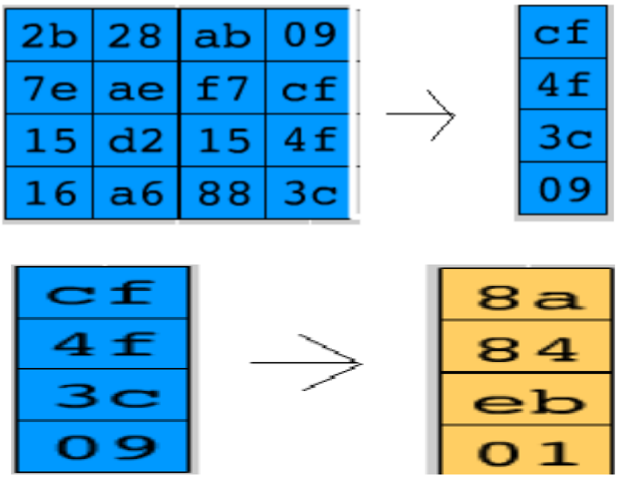
This can also be seen as the following:

$$\begin{aligned} b_0 &= 2a_0 + 3a_1 + 1a_2 + 1a_3 \\ b_1 &= 1a_0 + 2a_1 + 3a_2 + 1a_3 \\ b_2 &= 1a_0 + 1a_1 + 2a_2 + 3a_3 \\ b_3 &= 3a_0 + 1a_1 + 1a_2 + 2a_3 \end{aligned}$$

Where "+" xor operation.

Figure 4: Mixed Column Operation

5) *Key Expansion:* Generating all Round Keys from the initial input key is described by the key expansion phrase. In the event that the first round key is the original key, cryptography, and in the event that the final set of the Original keys are those produced by key expansion keys. Since as previously indicated, this first round key will be added to the input is required before encryption or decryption can begin. Ten sets of circular keys with a key size of 128 bits will be produced in a 16-byte size. Around the keys are produced word for word. The formula for producing the The round key's ten rounds are as follows: The fourth row each element is shifted up by rotating the i-1 key in a row.



2b	28	ab	09
7e	ae	f7	cf
15	d2	15	4f
16	a6	88	3c

cf
4f
3c
09

cf
4f
3c
09

8a
84
eb
01

Figure 5: Key Expansion

It then puts this result through a forwards Sub Box algorithm which replaces each 8 bits of the matrix with a corresponding 8-bit value from S-Box. To generate the first column of the ith key, this result is XORed with the first column of the i-1th key as well as a constant (Row constant or Rcon) which is dependent on i. The second column is generated by XOR-ing the 1st column of the ith key with the second column of the i- 1th key. This continues iteratively for the other two columns in order to generate the entire ith key. Additionally this entire process continues iteratively for generating all 10 keys. As a final note, all of these keys are stored statically once they have been computed initially as the ith key generated is require for the (10-i)th round of decryption.

V. Working of the AES System

The AES works systematically in the following manner:

- Step 1: We take word input (the data to be encrypted) and Key input (code word).
- Step 2: We perform Add Round Key operation.
- Step 3: We then perform Key Expansion.
- Step 4: We move on to performing Round Operation nine times.
- Step 5: Lastly, the Round Operation is executed one last time, excluding the Mixed Column Operation. It is thus called Last Round Operation.

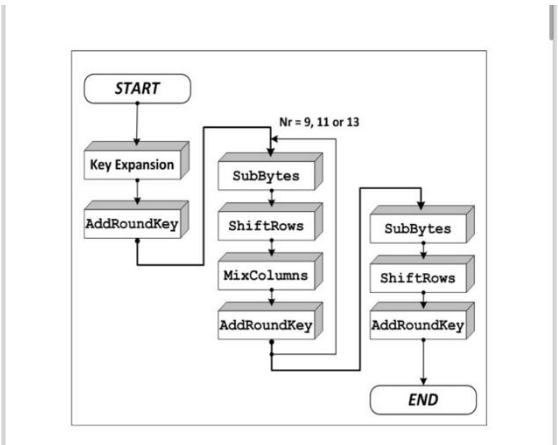


Figure 6: Working of AES encryption algorithm

AES-128 is widely employed in various security-sensitive applications, including online communications, secure file storage, and network security. Its robust security features make it a suitable choice for protecting sensitive data.

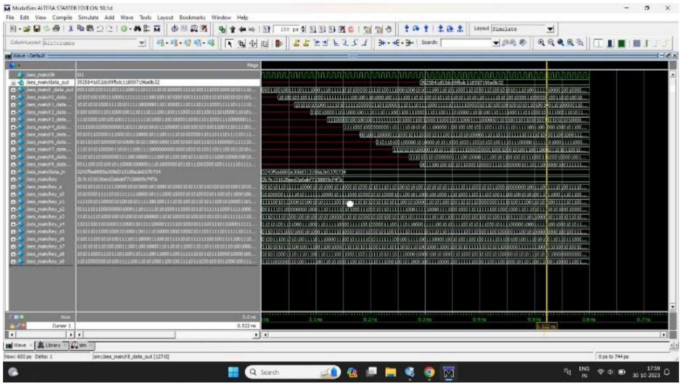


Figure 7: Simulation Results

VI. Implementation of AES Algorithm Using FPGA & Its Performance Analysis

The Advanced Encryption Standard (AES) has emerged as a widely accepted and secure symmetric encryption technique, offering high computational efficiency and versatility in various applications. This literature review delves into the extensive research conducted on optimizing AES implementations, particularly in the context of hardware architectures and algorithms. The review highlights key advancements, such as the development of optimized S-box and Mix Columns operations on dedicated ASIC and FPGA platforms, which enhance the security and speed of encryption/decryption. The potential for expanding key size and input block to increase security and throughput is also explored. With a focus on FPGA-based implementations, this survey identifies the evolving landscape of AES encryption techniques and underscores the promising opportunities for future work in multimedia communications and the support of multiple key lengths and modes of operation.

VII. Security Considerations:

While AES-128 is considered highly secure, the security of any cryptographic system depends on factors beyond the algorithm itself. These factors include key management, implementation security, and resistance to side-channel attacks.

VIII. Conclusion:

The AES-128 algorithm, as a part of the AES family, stands as a testament to the importance of robust encryption standards in today's digital world. Its 128-bit key length and efficient operations make it a popular choice for ensuring data confidentiality and integrity. However, it is essential to consider the broader security context when implementing AES-128 in real-world applications.

IX. References:

[1] IEEE [CCSP 2015, Area Optimized Implementation of AES Algorithm on FPGA, Hrushikesh S. Deshpande, Kailash J. Karande, AItaafO. Mulani, 978-1-4799-8081-9/15/$31.00 © 2015 IEEE.

[2] Implementation of AES Algorithm Using FPGA & Its Performance Analysis, Sonali A. Varhade1 , N. N. Kasat, ISSN (Online): 2319-7064, Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

[3] An FPGA Implementation of the AES with Fault Detection Countermeasure, Hassen Mestiri, Noura Benhadjyoussef, Mohsen Machhout and Rached Tourki, 978-1-4673-5549-0/13/$31.00 ©2013 IEEE.

[4] Efficient Implementation of AES Algorithm in FPGA Device, Swinder Kaur, Prof. Renu Vig, 0-7695-3050-8/07 $25.00 © 2007 IEEE DOI 10.1109/ICCIMA.2007.250

[5] Design and Implementation of Advanced Encryption Standard, K. MUKESH NAIDU, B. DEVA KUMAR, P.N. VAMSHI, INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

[6] Design and Implementation of Advanced Encryption Standard Security Algorithm using FPGA, Adnan Mohsin Abdulazeez, Ari Shawkat Tahir, International Journal of Scientific & Engineering Research, Volume 4, Issue 9, September-2013 1988 ISSN 2229-5518

[7] Youtube Video Link:
https://youtu.be/YVT4fcW7sI8?si=_Yn0LwoJ49vcGDkH