Design and Verification of AES encryption with new approach for S-Box

Bhupendra Badoniya, Ravi Mohan

Abstract— Information security has become a very critical aspect of modern computing systems. With the global acceptance of internet, virtually every computer in the world today is connected to every other. While this has created tremendous productivity and unprecedented opportunities in the world we live in, it has also created new risks for the users of these computers [2]. The users, businesses and organizations worldwide have to live with a constant threat from hackers and attackers, who use a variety of techniques and tools in order to break into computer systems, steal information, change data and cause havoc [2]. The paper work aims at designing and implementing a secure data communication between any two users based on the realization of advanced Symmetric-key Cryptographic algorithm called Advanced Encryption Standard (AES) on an FPGA based processor.

Index Terms— AES (Encryption & Decryption) Algorithm, CPLD (Complex Programmable Logic Design), EDA (Electronic Device Automation), FPGA (Field Programmable Gate Array), ISE (Integrated Simulation Environment), IOB (Input Output Buffer), LUT (Look up table).

1 Introduction

Cryptographic technology is an important way to ensure information security, and is the key to information safety. Among all kinds of cryptographic algorithms, Advanced Encryption Standard Algorithm (AES) is preferred as it offers high security, efficiency, convenient usage, flexibility, and comprehensive performance [4].

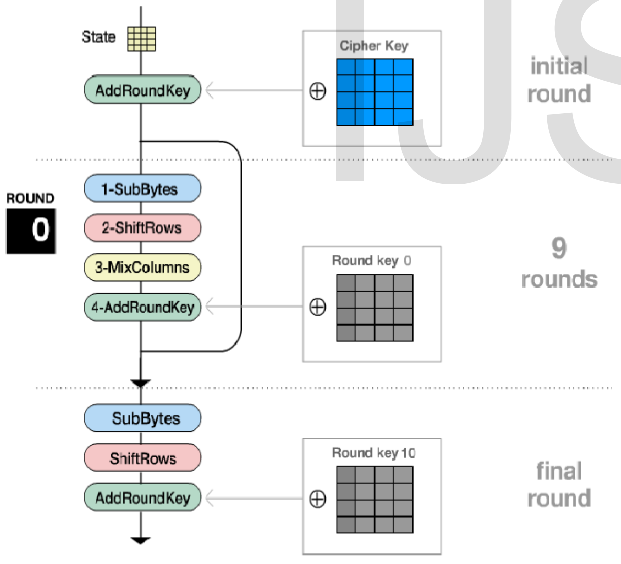


Fig 1. Encryption process Block Diagram

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 the blocks of bits[7]. AES cipher is specified as a number of repetitions of transformation rounds that convert the input plaintext into the final output of ciphertext.

Each round consists of several processing steps, including one that depends on the encryption key. A set of reverse rounds are applied to AES is based on a design principle known as a Substitution permutation network. Unlike its predecessor, DES, AES does not use a Feistel network. AES operates on a 4 x 4 array of bytes called state in a matrix form. The algorithm consists of performing four different simple operations. These operations are: Sub Bytes, Shift Rows, Mix Columns and Add Round Key.

2 Design Method

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

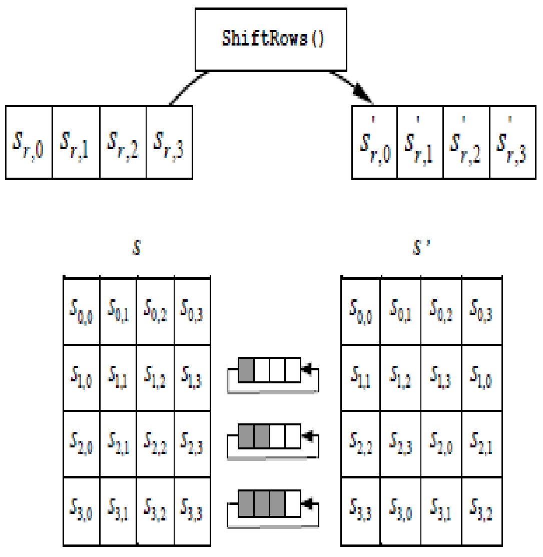
2.1 SubBytes transformation: is a non-linear byte substitution that operates independently on each byte of the State using a substitution table (S-box)[1].

- *Bhupendra Badoniya is currently pursuing masters degree program in electronics and communication engineering in RGPV University, India, PH-08359018239. E-mail: bhupendra.badoniya@gmail.com*
- *Ravi Mohan is currently working in Electronics & Communication department in SRIT, Jabalpur RGPV University, India, PH-09406737876. E-mail: ravimohan7677@yahoo.co.in*

TABLE 1
S-BOX: SUBSTITUTION

		y															
		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
x	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

2.2 SHIFTROWS TRANSFORMATION: The first row, r = 0, is not shifted. The shift value shift (r, Nb) depends on the row number, r, as follows (recall that Nb = 4):


ShiftRows ()
$s_{r,0}$ $s_{r,1}$ $s_{r,2}$ $s_{r,3}$
$s'_{r,0}$ $s'_{r,1}$ $s'_{r,2}$ $s'_{r,3}$
s
s'

shift(1,4) =1; shift(2,4) = 2 ; shift(3,4) = 3
Fig 2. ShiftRows ( ) cyclically shifts the last three rows in the state.

2.3 SHIFTROWS TRANSFORMATION: The Mix-Columns() transformation operates on the State column-by-column, treating each column as a four-term polynomial. In the MixColumns step, each column of the state is multiplied with a fixed polynomial a(x).

In the MixColumns step, the four bytes of each column of the state are combined using an invertible linear transformation. The MixColumns function takes four bytes as input and outputs four bytes, where each input byte affects all four output bytes.
Columns are considered with fixed

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} . \qquad (1)$$

Polynomial a(x), given by

Let $$s'(x) = a(x) \otimes s(x) : \qquad (2)$$

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

for 0 =c < Nb MixColumns( ) operates on the state column-by-column.

2.4 ADDROUNDKEY TRANSFORMATION: a Round Key is added to the output of MixColumn operation (state) by a simple bitwise XOR operation. For each round of operation, separate key is generated using Key Expansion.

3 KEY EXPANSION

Round keys are derived from the cipher key using Rijndael's key schedule. The AES algorithm takes the Cipher Key, K, and performs a Key Expansion routine to generate a key schedule. The Key Expansion generates a total of Nb (Nr + 1) words. The expansion of the input key into the key schedule proceeds as per the functions Rotword(), Subword(), Rcon[i/Nk], Xor operations[1].

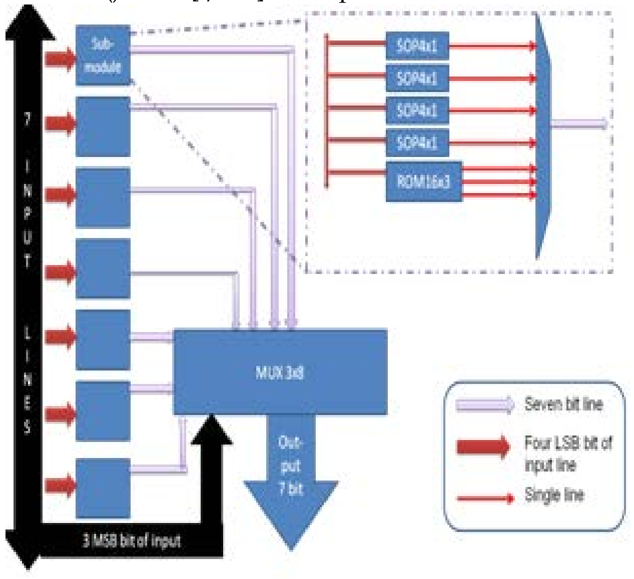


Fig 3. Proposed architecture S8 box

4 Tool Platforms and Language used

4.1 Tool: Xilinx ISE: It is a software tool produced by Xilinx for synthesis and analysis of HDL designs. Language used: Verilog HDL: Verilog, standardized as1364, is a description language (HDL) used to model electronic systems. It is most commonly used in the design and verification of circuit's the-transfer level [5].

4.2 platform used: Family Vertex4, Device- XC4VLX80, Package-FF1148. Target FPGA is a Vertex FGPA because the same platform is been used by base papers [5].

5 Simulation and synthesize of Proposed work

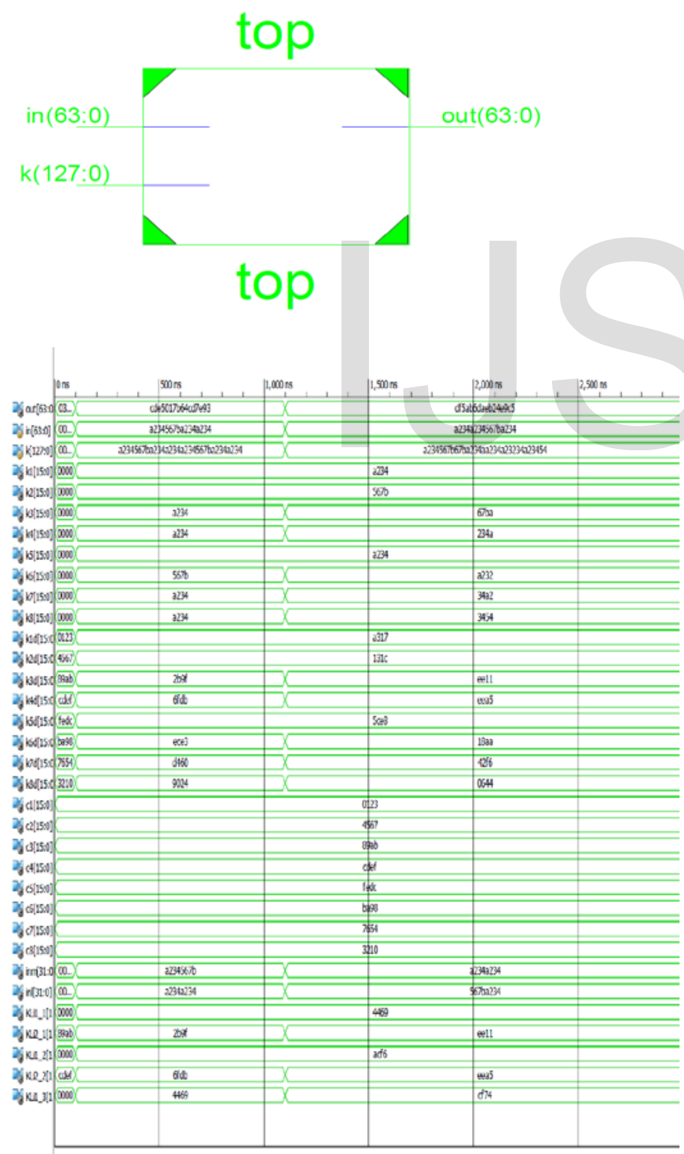


Fig 4. Simulation and RTL schematic of proposed work

6 Results

From the simulation as shown in above slides

Key : A234567ba234a234a234567ba234a234

Result:-1	
Output:	Cde5017b64cd7e93
Input:	A234567ba234a234
Output^Input:	6fd15700c6f9dca7
Avalanche:	41 bit change/64 bit
Result:-2	
Output:	Df5ab6daed24e9c5
Input:	A234a234567ba234
Output^Input:	7d6e14eebb5f4bf1
Avalanche:	45 bit change/64 bit

Table 2
Results for each module

	Parameters	Base [1]		Base[2]		Proposed work	
S-box design		S-box 7 (S7)	S-box 9 (S9)	S-box 7 (S7)	S-box 9 (S9)	S-box 7 (S7)	S-box 9 (S9)
	No. of slice	34	169	-	-	26	157
	Logical Time delay (ns)	-	-	-	-	6.067	7.279
Overall Kasumi encryption design	No. of slice	8784		8770		8401	
	Logical Time delay (ns)	34.01		-		33.64	

7 Comparative Results

Table 3
comparative results

Paramete rs	Design of FI	Design of FO	Design of FL	Design of Sbox-7	Design of Sbox-9	Complete KASUMI module
No. of slice	42 9	1379	18	26	157	8401
No. of LUT's	78 2	2541	32	52	289	15468
No. of IOB's	13. 04 ns	11.216 ns	4.303 ns	6.06 7 ns	7.279 ns	33.64 ns

8 Conclusions

The work is implemented of FPGA which makes proposed work a semicustom design as known semicustom design always lack behinds compare to full-custom design in term of Area, speed and power. In future proposed work can be implemented at transistor level (i.e. Full-custom).

Acknowledgment

I Bhupendra Badoniya would like to thanks to Prof Ravi Mohan for their valuable support and timely guidelines.

References

[1] Dr.R.V.Kshirsagar1, M.V.Vyawahare2, FPGA Implementation of High speed VLSI Architectures for AES Algorithm, 2012 Fifth International Conference on Emerging Trends in Engineering and Technology, 978-0-7695-4884-5/12, 2012 IEEE, DOI 10.1109/ICETET.2012.53

[2] Shylashree.N1, Nagarjun Bhat2 and V. Shridhar3, FPGA IMPLEMENTATIONS OF ADVANCED ENCRYPTION STANDARD: A SURVEY, International Journal of Advances in Engineering & Technology, May 2012. ISSN: 2231-1963

[3] Mr. Atul M. Borkar, Dr. R. V. Kshirsagar, Mrs. M. V. Vyawahare, FPGA Implementation of AES Algorithm, 978-1-4244-8679-3/11, 2011 IEEE

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

[5] *http://www.xilinx.com/support.html*

[6] Wisniewski, Remigiusz (2009). *Synthesis of compositional microprogram control units for programmable devices*. Zielona Góra: University of Zielona Góra. p. 153. ISBN 978-83-7481-293-1

[7] Thomas Jakobsen and Lars Knudsen. The interpolation attack on block ciphers. In *Fast Software Encryption '97*, volume 1267 of LNCS, pages 28–40. Springer-Verlag, 1997.

[8] Eli Biham and Adi Shamir. *Differential Cryptanalysis of the Data Encryption Standard*. Springer-Verlag, 1993.

[9] Mitsuru Matsui. Linear cryptanalysis method for DES cipher. In *Advances in Cryptology—Eurocrypt '93*, volume 765 of LNCS, pages 386–397. Springer-Verlag, 1993.