



DESIGN OF COMBINATIONAL S-BOX IMPLEMENTED AES ALGORITHM FOR MULTIPLE FAULT DETECTION SCHEME

V.Nanuku Naik¹, P. Michael Cholines²

¹Pursuing M.tech (VLSI), ²Assoc.Professor (ECE),

*Nalanda Institute of Engineering and Technology (NIET), Siddharth Nagar, Kantepudi (V),
Sattenpalli (M), Guntur Dist, A.P. (India)*

ABSTRACT

In this paper we are designing combinational S-box based AES. The Advance encryption is method of encrypting the data with n varied sizes. AES will have sub byte transformation as one of the step that will be performing to make encryption. The proposed method will have combinational logic based Substitution box, where in existing system will have S-box as a look up table such that it needs a memory. The sequential circuits are slower than the combinational designs. Since instead of the ROM based look up table we will implement it as a combinational design such that its having less complexity and high speed. All of the modules in the architecture are designed using Verilog HDL, it is synthesized in Xilinx ISE 13.2i.

Keywords: *S Box, Sub Byte, Shift Rows, Mix Columns, Add Round Key.*

I. INTRODUCTION

Encryption is a process of converting data from readable format to unreadable format. The readable format called as plain text. The unreadable data format we called as cipher text. The process used to convert the algorithm we call as cipher. There are especially two types of algorithms are existing are Data encryption standard and Advance encryption standard. Any encryption algorithm will comprise of data as one input, another parameter that is required to convert the data we call as key so we say that second input is key. The DES will have data size of 64 bit and key size of 56 bit. This is inefficient due to smaller size of data and key. It also has one more drawback of asymmetric key. In the year 1991 the NIST called up for the new encryption algorithms. They have conducted several examinations on the received encryption methods; among them they have selected 5 algorithms are RC6, Rijndael, Mars, Serpent and Two fish. They have subjected those algorithms to further rounds and they found Rijndael algorithm is the best algorithm. The scientist named Rijndael proposed an algorithm which is efficient than the DES. The AES will have data size and key size are variable i.e data size may be 128,196,256 etc, the key size may also varies with 128 , 194 and 256 bit. It is having certain stages all are performed in loop passion called as round. The algorithm will performed in certain rounds, the number rounds will be varied based size of the key that will be using in algorithm. That may be

10,12 and 14 rounds based on key size of 128,194,256 respectively. In each round it will perform four stages that might be sub byte transformation, shift rows , mix columns and add round key.

The advance encryption standard key size will affects the number of rounds that it will perform on the given data. The encryption will secure the data by making conversion to the unknown format. It is more secure due to its complex process. The software design of AES will be easy and less secure, that needs slowest process, to support high data rate, high bit width communications we need to implement it in hardware. The existed AES using look up table in the sub byte transformation, but in proposed method we are using combinational circuit based s-box make the system highly secure, speed and fault detectable.

II. AES ALGORITHM

The AES is advance encryption standard which was proposed by the scientist Rijndael. The encryption is the process of transforming data from known format to unknown format. That encrypted data can be understood only by the person who possesses special skill on cryptography. The Advance encryption standard will have input as data and key. It accepts data and key as block of 32 bits with minimum size of 128 bit and maximum size of 256bit. The AEs is more complex process such that it will perform certain loops. The AES is generally performed as set of four stages. The four stage cumulatively called as round. The number of rounds that are performing will be varied based on the key size used in the AES. The rounds may be 10,12,14 with corresponding key sizes are 128,196 and 256bits. The four stages are namely sub byte transformation, shift rows, mix columns and add round key. The final round will accomplish with only three stages remaining all rounds will contain all stages of AES. Each round will use distinct key in the AES. All several keys are generated by using Key expansion algorithm. The key expansion algorithm will supply all keys that are needed to add round key. In this paper we are concentration on 128 bit data and 128 key sizes. The data will be given to the first stage i.e sub byte transformation then transformed data will be given to the shift rows, then the shifted data will be giving to the mix column and finally the output data from mix columns will be given to the final round add round key where data and key will added. The 128 bit data is divided into 16 blocks each consist of one byte of data. The data 16 blocks are called as states. The 16 states are arranged in 4*4 matrixes. Complex mathematical operation will be performed on the data at each stage of the AES. The all operations are Galois's field operations.

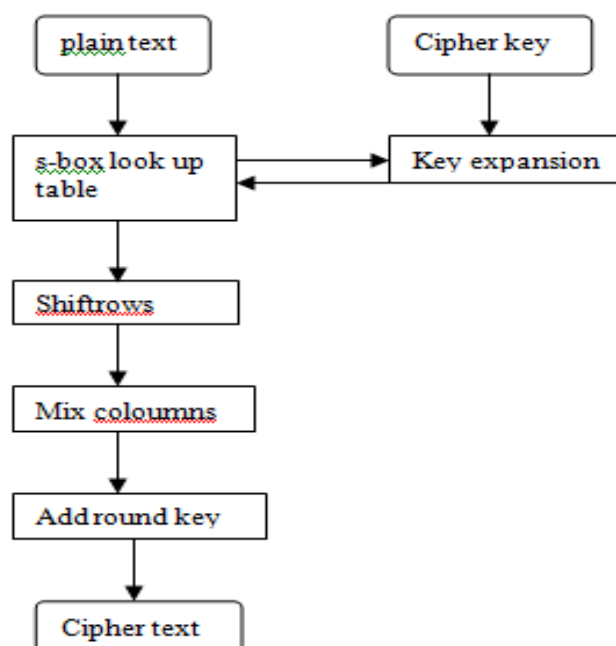


Fig:1 Existing flow chart for AES algorithm

2.1 Sub Byte Transformation

This is the first and foremost step in the AES. In this step each byte in the input data will be replaced with pre calculated data from the pre calculated table this will be continued until completion of all states. That table we call as S-box which contains all values from 0 to 256 and its associated values of calculation. This S-box is implemented as Look up table Rom which makes delay of the circuit. It is the nonlinear step, i.e each byte is individually replaced irrespective of the other byte

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	1	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	4	C7	23	C3	18	96	5	9a	7	12	80	E2	eb	27	B2	75
	4	9	83	2c	1a	1b	6e	5a	A0	52	3b	D6	B3	29	E3	2f	84
	5	53	D1	0	Ed	20	Fc	B1	5b	6a	Cb	Be	39	4a	4c	58	cf
	6	D0	Ef	Aa	Fb	43	4d	33	85	45	F9	2	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	29	90	88	46	Ee	B8	14	de	5e	0b	db
	a	E0	32	3a	0a	49	6	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	8
	c	Ba	78	25	2e	1c	A6	B4	C6	E8	Dd	74	1f	4b	bd	8b	8a
	d	70	3e	B5	66	48	3	F6	0e	61	35	57	ba	86	C1	1d	9e
	e	E1	F8	98	11	69	D9	8e	94	9b	1e	87	ca	ce	55	28	df
	f	sc	A1	89	0d	bf	E6	42	68	41	99	2d	0f	B0	54	bb	16

Fig:2 S Box Table

2.2 Proposed Design of Aes

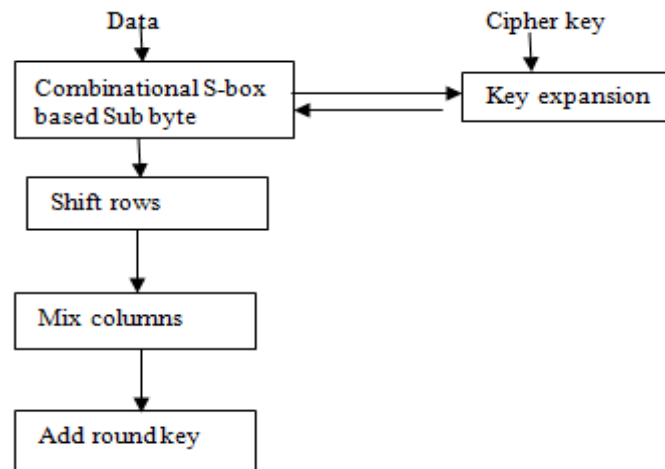


Fig: 3proposed design flow chart

The proposed advanced encryption standard will also have the same stages as presented in the existing system except for the small change in the sub byte transformation stage. This is also used in the key expansion algorithm. In sub byte transformation, we have a pre-calculated table which is actually implemented as a ROM Look up table in existing designs. The look up table will have an unbreakable delay, it will not support pipelining and it has the possibility that hackers may hack the S-box such that we can identify errors in it.

To avoid all these problems, we are implementing a hardware level S-box design using a combinational structure. To make the transformation, we actually perform multiplicative inversion and affine transformation to get the resultant values. Now we have to realize them on the input data. Data input for the S-box is eight bits, output is also eight bits, i.e., one byte. The hardware realization of the S-box is shown in the block diagram. We have implemented a combinational S-box as five important blocks. Among them, the first three blocks perform multiplicative inversion. The last two blocks will perform affine transformation. It also includes a parity mechanism to avoid faults in the design. It will perform transformation on a single byte at a time such that the input will be 8 bits. The 8-bit input will be given to the first block called isomorphic transformation, which converts data from binary to Galois field such that all calculations can be performed very easily. Then they will be divided into four bits and fed to the XORing block and squaring block. The squaring block performs squaring of the taken data. Then it will multiply with a constant λ . Lower bits of the data will be XORed with upper bits, and then the result will be added to the product at the λ .

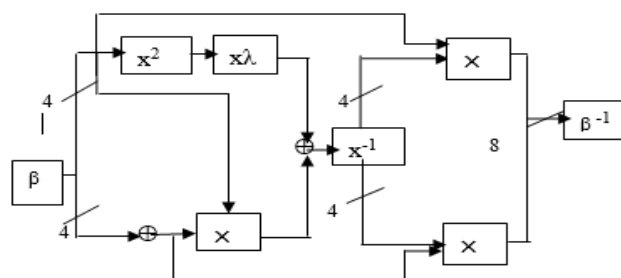


Fig 4Block diagram of s-box

β -----isomorphic transformation
X----- Multiplication
 $X\lambda$ -----multiplication with lambda
 x^2 -----Squarer
 δ^{-1} -----inverse isomorphic transformation
 X^{-1} ----- Multiplicative inversion

Where q is 8 bit incoming data

$$\beta \times q = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \times \begin{pmatrix} q7 \\ q6 \\ q5 \\ q4 \\ q3 \\ q2 \\ q1 \\ q0 \end{pmatrix}$$

$$\beta^{-1} \times q = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} q7 \\ q6 \\ q5 \\ q4 \\ q3 \\ q2 \\ q1 \\ q0 \end{pmatrix}$$

Fig: 5 Isomorphic transformations, Fig: 6 Inv isomorphic transformations

2.3 Isomorphic Transformation

To perform multiplicative inversion first we have to split the data from higher order bits to lower order bits such as GF(28) into GF(2¹), GF(2²)...etc. To accomplish multiplicative inversion we also need conversion from composite field to Galois field such that it will be very easy to perform operation in Galois field. The isomorphic transformation is presented in matrix form as shown in below. That was denoted by δ is an 8*8 matrixes and multiplication is AND operation, addition is modulo-2 addition.

Where q is 8 bit incoming data

$$\delta \times q = \begin{pmatrix} q^7 \oplus q^5 \\ q^7 \oplus q^6 \oplus q^4 \oplus q^3 \oplus q^2 \oplus q^1 \\ q^7 \oplus q^5 \oplus q^3 \oplus q^2 \\ q^7 \oplus q^5 \oplus q^3 \oplus q^2 \oplus q^1 \\ q^7 \oplus q^5 \oplus q^3 \oplus q^2 \\ q^7 \oplus q^6 \oplus q^2 \oplus q^1 \\ q^7 \oplus q^4 \oplus q^3 \oplus q^2 \oplus q^1 \\ q^6 \oplus q^4 \oplus q^1 \\ q^6 \oplus q^1 \oplus q^0 \end{pmatrix}$$

$$\delta^{-1} \times q = \begin{pmatrix} q^7 \oplus q^6 \oplus q^5 \oplus q^1 \\ q^6 \oplus q^2 \\ q^6 \oplus q^5 \oplus q^1 \\ q^6 \oplus q^5 \oplus q^4 \oplus q^2 \oplus q^1 \\ q^5 \oplus q^4 \oplus q^3 \oplus q^2 \oplus q^1 \\ q^7 \oplus q^4 \oplus q^3 \oplus q^2 \oplus q^1 \\ q^5 \oplus q^4 \\ q^6 \oplus q^5 \oplus q^4 \oplus q^2 \oplus q^0 \end{pmatrix}$$

Fig 7. Isomorphic and inv isomorphic equations

2.4 Affine Transformation

The affine transformation which will performed after multiplicative inversion to accomplish the sub byte transformation, the sub byte transformation output will same as the pre calculated value after affine transformation. AT stands for affine transformation, the incoming data is denoted by “a”.

$$AT(a) = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} a_7 \\ a_6 \\ a_5 \\ a_4 \\ a_3 \\ a_2 \oplus \\ a_1 \\ a_0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

$$AT^{-1}(a) = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \times \begin{pmatrix} a_7 \\ a_6 \\ a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{pmatrix} \oplus \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$$

Fig: 8 Affine transformations.

2.5 Shift rows

In this step the input will be taken from the sub byte. The data which is arranged in the states, the rows of the states are shifted cyclically to little number of bits to the left. The first row of states will be not be changed. The second has to of states has to be shifted by one bit, third and fourth rows will be shifted accordingly three two and three bits. Inverse shift rows will accomplish with rows are cyclically shifted few number of bits to the right. The first row is an unaltered. The second row will be shifted by one bit successively all rows will be shifted by two and three bits.

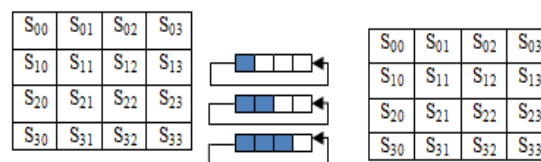


Fig: 9 Shift rows implementation

2.6 Mix Columns

In this stage input is taken from shift rows that will process as output. Each output in the mix column will be product of column of state matrix with the constant polynomial of Galois field. This design is actually implemented with module shown in figure below. The module contains combination of X time block followed by Exoring logic. The Xtime block which consists of combination of Xor logics with logical shift operations. The whole module works for single state. To accomplish it for all states we have to use four Xtime blocks in the design.

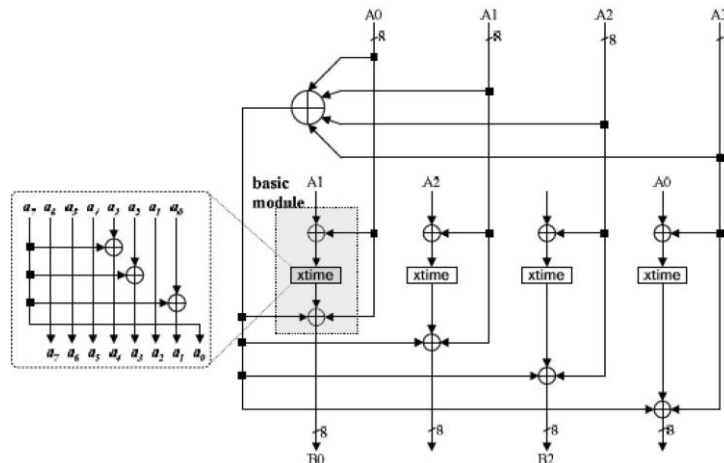


Fig: 10 Mix Column Implementation

2.7 Add Round Key

In this step the Mix column stage output will be fed back as input to it. The key is taken from the key expansion algorithm which is generated from the initial key of cipher. This adding is done by using Xoring operation between the individual byte to byte operation of the data and key.

2.8 Key Expansion

In the key expansion algorithm, input key is taken from the cipher key i.e initial key. That will be expanded to get different keys, this is usually done in several steps are initially it is taking 4 word and produces 44 words. It will be taking first four words 44 words. The rot word will be produced by making one byte cyclic left shift on the selected four words. Sub word will be produced by making substitution by te transformation using S-box. That will be Xored with the constant word, the constant word will have upper three bytes will be zeros. For the first phase Xoring will be done only to the least significant byte.

IV. SIMULATION RESULTS

In this project we are implementing novel combinational S-box based implementation of AES.The combinational implementation consists of five different modules which will perform multiplicative inversion and affine transformation. The Combinational design will yields parallel pipelined implementation with reduced complexity hardwareand small area occupancy. All of the combinational modules are designed using Verilog HDL. They are synthesized in Xilinx ISE 13.2i and also simulated in the same. The synthesis and simulationresults are as follows

Internal architecture for top module

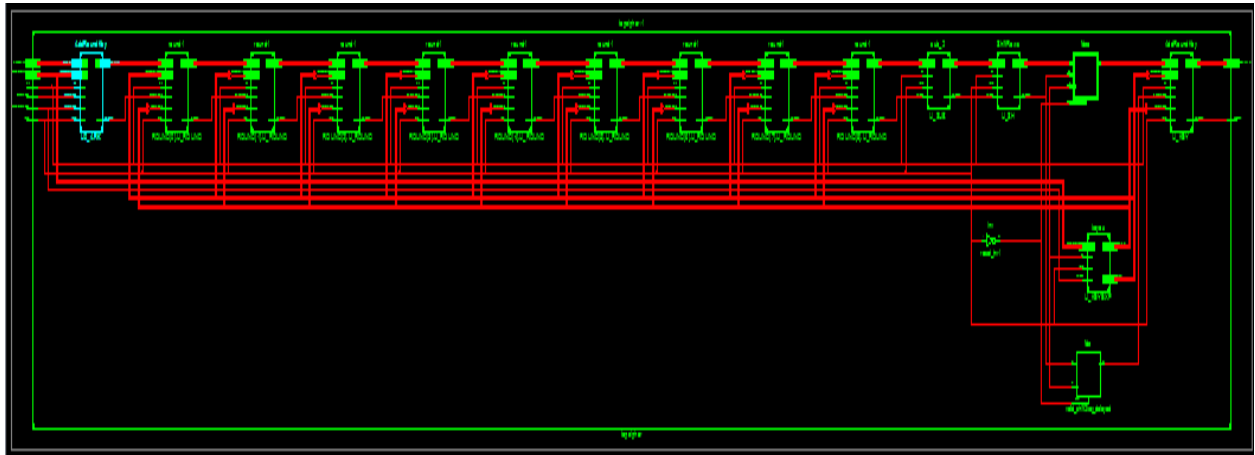


Fig: 11 Gate level model for top module

V. SYNTHESIS REPORT

topcipher Project Status (11/21/2015 - 15:28:32)					
Project File:	aesnew.xise	Parser Errors:	No Errors		
Module Name:	topcipher	Implementation State:	Synthesized		
Target Device:	xc7a30t-3csg324	Errors:	No Errors		
Product Version:	ISE 13.2	Warnings:	22 Warnings (22 new)		
Design Goal:	Balanced	Routing Results:			
Design Strategy:	Xilinx Default (unlocked)	Timing Constraints:			
Environment:	System Settings	Final Timing Score:			

Device Utilization Summary (estimated values)				
Logic Utilization	Used	Available	Utilization	
Number of Slice Registers	9169	42000	21%	
Number of Slice LUTs	18512	21000	88%	
Number of fully used LUT-FF pairs	5131	22550	22%	
Number of bonded IOBs	389	210	185%	
Number of BUFG/BUFGCTRLs	2	32	6%	

Detailed Reports					
Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	Sat 21. Nov 15:28:31 2015	0	22 Warnings (22 new)	0

Fig: 12 Area report

Timing constraint: Default period analysis for Clock 'clk'

Clock period: 4.896ns (frequency: 204.236MHz)

Total number of paths / destination ports: 19877435 / 17743

Delay: 4.896ns (Levels of Logic = 8)

Source: U_KEYEXP/RKGEN_U0/Key_FirstStage_29 (FF)

Destination: U_KEYEXP/RKGEN_U0/round_key_delayed_103 (FF)

Source Clock: clk rising

Destination Clock: clk rising

Data Path: U_KEYEXP/RKGEN_U0/Key_FirstStage_29 to U_KEYEXP/RKGEN_U0/round_key_delayed_103

Cell:in->out	fanout	Gate Delay	Net Delay	Logical Name (Net Name)
FDCE:C->Q	12	0.361	0.607	U_KEYEXP/RKGEN_U0/Key_FirstStage_29 (U_KEYEXP/RKGEN_U0/Key_FirstStage_29)
LUT4:I0->O	14	0.097	0.355	U_KEYEXP/RKGEN_U0/SUB_U/_i000001/d/Mxor_s<5>_xo<0>
LUT4:I3->O	3	0.097	0.521	U_KEYEXP/RKGEN_U0/SUB_U/_i000001/d4/n3/x[1]_y[1]_
LUT6:I3->O	8	0.097	0.543	U_KEYEXP/RKGEN_U0/SUB_U/_i000001/Mxor_u8_2_xo<0>
LUT4:I1->O	4	0.097	0.309	U_KEYEXP/RKGEN_U0/SUB_U/_i000001/d5/Mxor_v2<3>_xo
LUT5:I4->O	3	0.097	0.305	U_KEYEXP/RKGEN_U0/SUB_U/_i000001/d7/Mxor_r8_1_xo<
LUT6:I5->O	4	0.097	0.525	U_KEYEXP/RKGEN_U0/SUB_U/_i000001/d7/Mxor_r8_1_xo<
LUT4:I1->O	17	0.097	0.586	U_KEYEXP/RKGEN_U0/SUB_U/_i000001/d8/Mxor_p<7>_xo<
LUT6:I3->O	1	0.097	0.000	U_KEYEXP/RKGEN_U0/temp_round_key<99>1 (U_KEYEXP/RKGEN_U0/temp_round_key<99>1)
FDCE:D		0.008		U_KEYEXP/RKGEN_U0/round_key_delayed_99

Total 4.896ns (1.145ns logic, 3.751ns route)

(23.4% logic, 76.6% route)

Fig: 13 Time delay report

5.1 Simulation Result



Fig: 14 Simulation Result

VI. CONCLUSION

In this paper we have analyzed and designed Combinational S-box based AES using Verilog HDL. Individual and complete modules are synthesized and simulated in Xilinx ISE 13.2i. The proposed design of AES will be having less reduced complexity with compared to the Existed AES. It also supports parallel pipeline to increase the throughput of the Encryption process such that proposed idea of s-box will support it. The Proposed design having high security and fault detection is possible.

REFERENCES

- [1] Hoang Trang, Nguyen Van Loi "An efficient FPGA implementation of the AdvancedEncryption Standard algorithm"
- [2] Daemen J., and Rijmen V, "The Design of Rijndael: AES-the AdvancedEncryption Standard", Springer-Verlag, 2002
- [3] FIPS 197, "Advanced Encryption Standard (AES)", November 26, 2001.
- [4] Tessier, R., and Burleson, W., "Reconfigurable computing for digitalsignal processing: a survey", J.VLSI Signal Process., 2001, 28, (1-2),pp.7-27.

- [5] 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, 2010.
- [6] Alex Panato, Marcelo Barcelos, Ricardo Reis, "An IP of an Advanced Encryption Standard for Altera Devices", SBCCI 2002, pp. 197-202, Porto Alegre, Brazil, 9 and 14 September 2002.
- [7] Mr. Atul M. Borkar, Dr. R. V. Kshirsagar and Mrs. M. V. Vyawahare, "FPGA Implementation of AES Algorithm", International Conference on Electronics Computer Technology (ICECT), pp. 401-405, 2011 3rd
- [8] J. Vijaya and M. Rajaram, "High Speed Pipelined AES with Mix Column Transform," European Journal of Scientific Research, ISSN 1450-216X Vol.61 No.2 (2011), pp. 255-264.
- [9] Priyanka Pimpale, Rohan Rayarikar, Sanket Upadhyay, "Modifications to AES Algorithm for Complex Encryption," IJCSNS International Journal of Computer Science and Network Security, Vol.11 No.1 0, October 2011.
- [10] Aluned. H. Sawahneh, "Hardware Design of AES S-box using pipelining structure over GF((24i)".
- [11] K.Rahimwmsa, Dr. S. Sureshkumar, and K.Rajeshkumar, "Implementation of AES with New S-Box and Performance Analysis with the Modified S-Box," International Conference on VLSI, Communication & Instrumentation (ICVCI) 20J J Proceedings published by International Journal of Computer Applications® (IJCA).
- [12] MooSeop Kim, Juhan Kim, and Yongje Choi, "Low Power Circuit Architecture of AES Crypto Module for Wireless Sensor Network," World Academy of Science, Engineering and Technology 8, 2007.
- [13] M.Pitchaiah, Philemon Daniel, and Praveen, "Implementation of Advanced Encryption Standard Algorithm," International Journal of Scientific & Engineering Research, Volume 3, Issue 3, March -2012 ISSN 2229-5518.
- [14] Zine El Abidine, Alaoui Ismaili, and Aluned MOUSSA, "Self-Partial and Dynamic Reconfiguration Implementation for AES using FPGA," IJCSI International Journal of Computer Science, Issues, Vol. 2, 2009 ISSN (Online):1694-0784 ISSN (Print): 1694-081456

AUTHOR DETAILS

	V. NANUKU NAIK pursuing is M.tech in VLSI system design from Nalanda institute of Engineering and technology. He completed his B.tech in Electronics and Communication Engineering. His research of interest includes VLSI, CMOS Analog, Digital system design etc.
	P. MICHAEL CHOLINES is working as associate professor in Nalanda Institute of Engineering and technology. He completed his post-graduation in VLSI and his area of interest includes Digital CMOS, CMOS mixed signals etc.