International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 3 Issue 1 ǁ Jan. 2015 ǁ PP.09-14

Implementation and Design of AES S-Box on FPGA

Chandrasekhar Savalam[1] & Prasanti Korapati[2]
Assistant professor, ECE Dept., DIET College & Assistant professor, EIE Dept., VRSEC College

Abstract— The Advanced Encryption Standard can be programmed in software or built with pure hardware. However Field Programmable Gate Arrays (FPGAs) offer a quicker, more customizable solution. This research investigates the AES algorithm with regard to FPGA and the Very High Speed Integrated Circuit Hardware Description language (VHDL). Xilinx Design Suite 14.5 software is used for simulation and optimization of the synthesizable VHDL code. All the transformations of both Encryptions is simulated using an iterative design approach in order to minimize the hardware consumption. Virtex 6 Family devices are utilized for hardware evaluation. Advanced Encryption Standard (AES) is one of the most common symmetric encryption algorithms. The hardware complexity in AES is dominated by AES substitution box (S-box) which is considered as one of the most complicated and costly part of the system because it is the only non-linear structure. Theoretically, the design reduces the overall delay and efficiently for applications with high-speed performance. This approach is suitable for FPGA implementation in term of gate area. The hardware, total area and delay are presented.
Keywords—S-Box, VHDL, AES, FPGA

I. INTRODUCTION

This AES algorithm is based on Rijndael algorithm,which was developed by Joan Daemen and VincentRijnmen. It was announced by National Institute of Standards and Technology (NIST) in 1997 as AES algorithm [1] according to the primary criteria of security, performance, efficiency in software and hardware, flexibility, and implementability.

One of the strength of AES is it simplicity which facilitates the implementation on a wide range of different platforms under different constraints [2]. AES can be implemented using software and hardware. Hardware implementation can be either based on Field Programmable Gate Array (FPGA) or Application Specific Integrated Circuits (ASIC). AES implemented on FPGA offered flexibility and security, but medium speed compared to ASIC. The major factors that influence the implementation choices are speed and area cost. By using hardware, a higher data rate for fast applications such as routers can be achieved compared to software implementation. The hardware implementation is also physically secure since tempering by an attacker is difficult. The efficiency of AES hardware implementation in terms of size, speed, security and power consumption depends largely on the AES architecture [3]. For high speed throughput, loop-unrolled pipelined [4] is used and to save power and area, iterative single round with resource sharing is implemented.

As S-Box is considered as a full complexity design and causes high power dissipation in AES, this paper is focused on the way to implement it efficiently. Traditionally it was implemented by look up tables (LUT) which store all 256 bit predefined values of S-Box in a ROM. The advantage of using LUT is it offers a shorter critical path. However, it has a drawback of the unbreakable delay in high speed pipelined designs, and hence it cannot be used in high speed applications. This delay prohibits each round unit from being divided into more than two sub-stages to achieve any further increase in processing speed [5]. It also requires a large area to implement AES encryption and decryption system due to different table used for both systems. Another way is to design the S-Box circuit using combinational logic only which is directly from its arithmetic properties. This approach has breakable delay of S-Box processing. One of the S-Box design is from its truth table using direct relationship between input and output values of the S-box and two-level logic, which make use of sum of products (SOP) expression, product of sums (POS) expression, positive polarity Reed-Muller (PPRM) structure, binary decision diagram (BDD) or its variance twisted binary decision diagram (TBDD). It can achieve a high speed design but suffer from extremely large area cost. Rashmi et al. [6] implemented S-Box using a combinational logic without involving the computation in Galois field. He proposed two techniques for implementing the S-Box, (1) based on logic synthesis using truth table, and (2) direct implementation of the Algebraic Normal Form (ANF) expression for each column. It claims to have very low critical path delay compared to designs based on composite field. Other approach involves multiplicative inversion in Galois Field GF 2^8 using composite field [7]. It leads to low area but the critical delay is increased and it has high hardware complexities. In the next section, the AES algorithm is described briefly. In section III, it includes the discussion on the Sub Byte transformation. Subsequently, in section IV it includes the results and comparisons.

II. ENCRYPTION PROCESS

The Encryption process of Advanced Encryption Standard algorithm is presented below, in figure 1.

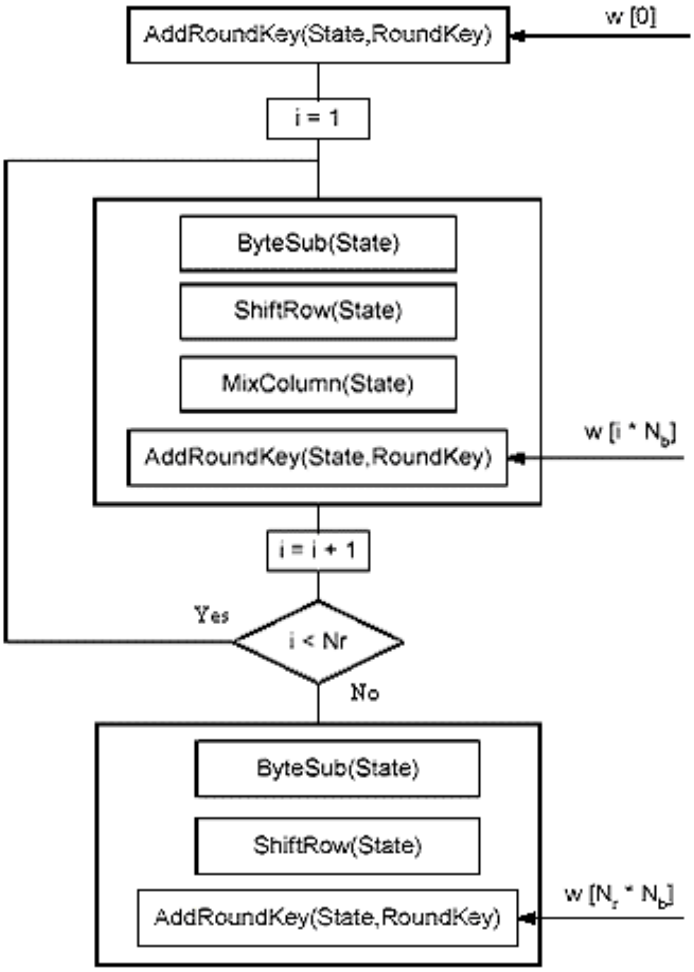

Figure 1. Encryption Process

This block diagram is generic for AES specifications. It consists of a number of different transformations applied consecutively over the data block bits, in a fixed number of iterations, called rounds. The number of rounds depends on the length of the key used for the encryption process.

III. BYTES SUBSTITUTE TRANSFORMATION

The bytes substitution transformation Byte sub (state) is a non-linear substitution of bytes that operates independently on each byte of the State using a substitution table (Sbox) presented in figure7. This S-box which is invertible, is constructed by composing two transformations

1. Take the multiplicative inverse in the finite field GF (2^8), described in Section 1.3.2. The element {00} is mapped to itself.
2. Apply the following affine transformation (over GF (2))

$$b_i^{'} = b_i \oplus b_{(i+4) mod\ 8} \oplus b_{(i+5) mod\ 8} \oplus b_{(i+6) mod\ 8} \oplus b_{(i+7) mod\ 8} \oplus c_i$$

for $0 \leq i \leq 8$, where b_i is the i^{th} bit of the byte, and c_i is the i^{th} bit of a byte c with the value {63} or {01100011}. Here and elsewhere, a prime on a variable (e.g., b') indicates that the variable is to be updated with the value on the right. In matrix form, the affine transformation element of the S-box can be expressed as

$$\begin{bmatrix} b_0' \\ b_1' \\ b_2' \\ b_3' \\ b_4' \\ b_5' \\ b_6' \\ b_7' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Figure 2. Matrix Notation of S-box

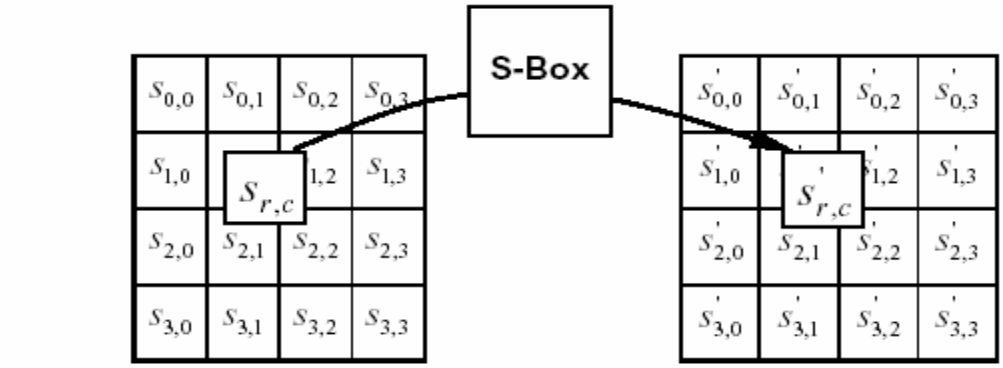

Figure 3. Application of S-box to the Each Byte of the State [1]

The S-box used in the Sub Bytes transformation is presented in hexadecimal form in figure 3. For example, if =S1,1= {53}, then the substitution value would be determined by the intersection of the row with index '5' and the column with index '3' in figure 3. This would result in S'1, 1 □ having a value of {ed}.

		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	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 4. S-box Values for All 256 Combinations in Hexadecimal Format [1]

IV. RESULT

VHDL is used as the hardware description language because of the flexibility to exchange among environments. The code is pure VHDL that could easily be implemented on other devices, without changing the design. The software used for this work is Xilinx Design Suite 14.5. This is used for writing, debugging and optimizing efforts, and also for fitting, simulating and checking the performance results using the simulation tools available on Xilinx 14.5 design software. All the results are based on simulations from the Xilinx ISim Simulator, using Timing Analyzer and Waveform Generator. All the individual transformation of encryption is simulated using FPGA Vertex 6 family and xc6vlx240t devices. The characteristics of the devices are presented in figure 17. An iterative method of design is implemented to minimize the hardware utilization and the fitting is done by the Xilinx Design Suite 14.5

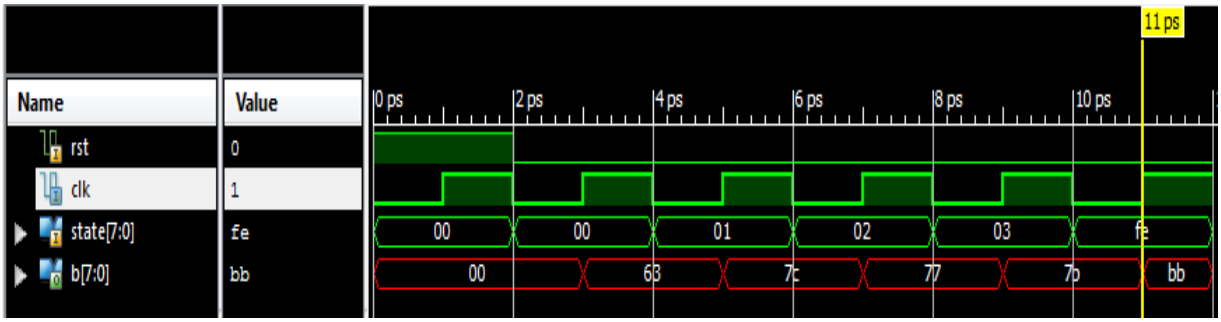

Figure 5. Waveform of SubByte Transformation

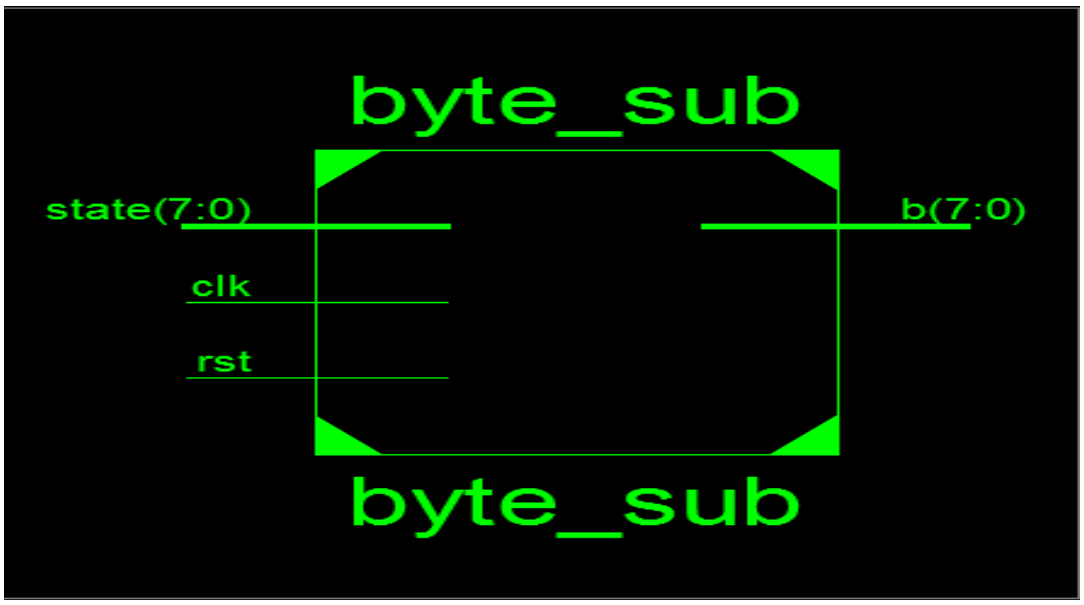

Figure 6. RTL Schematic of SubByte Transformation

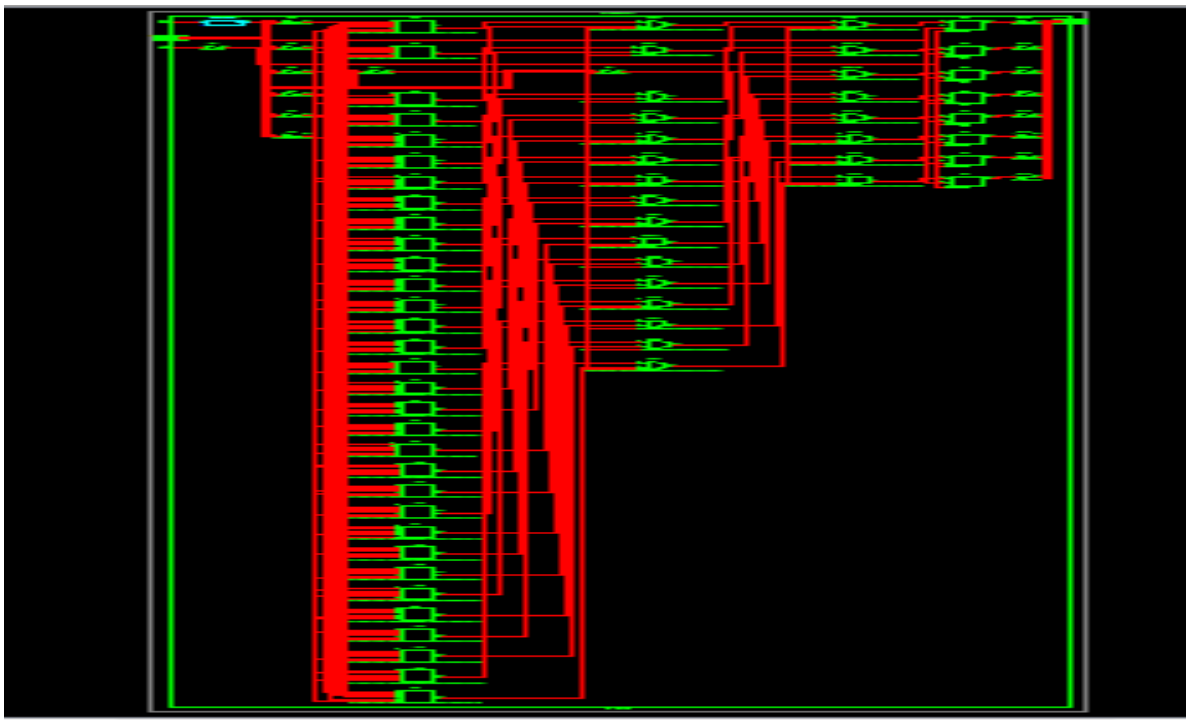

Figure 7. Array Structure Of SubByte Transformation

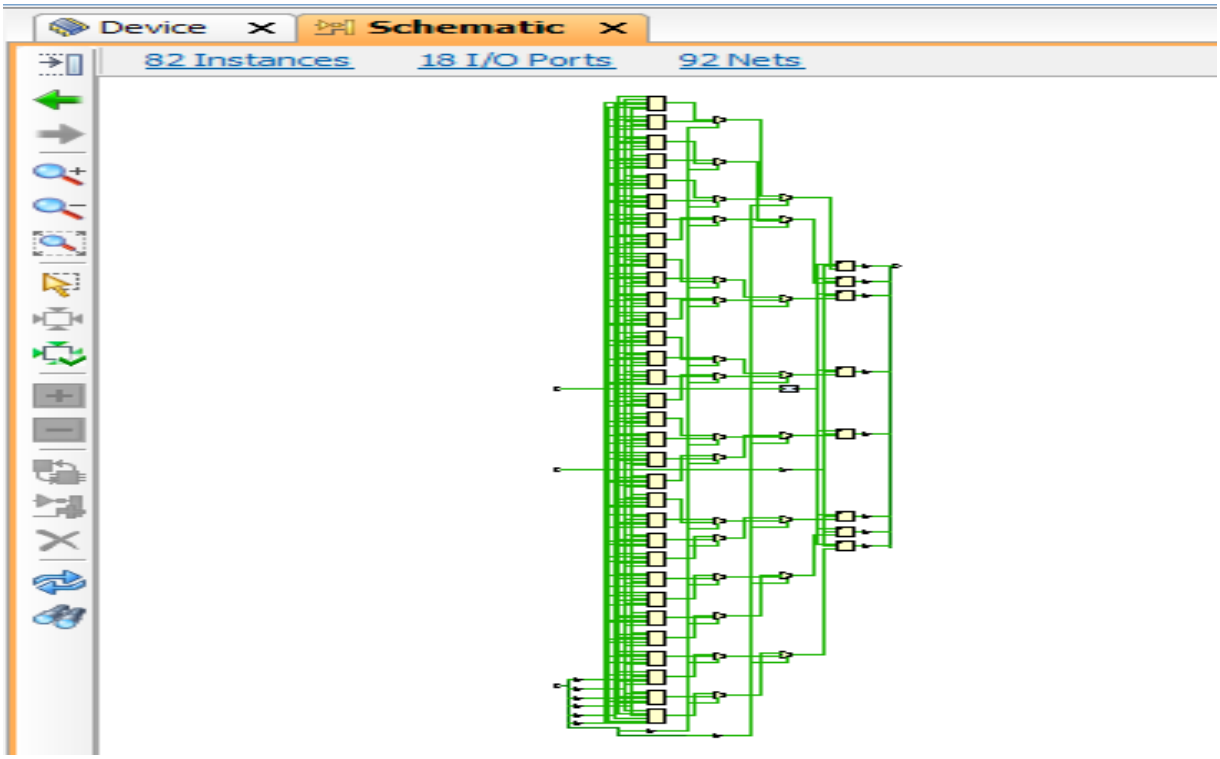

Figure 8. Schematic of SubByte Transformation

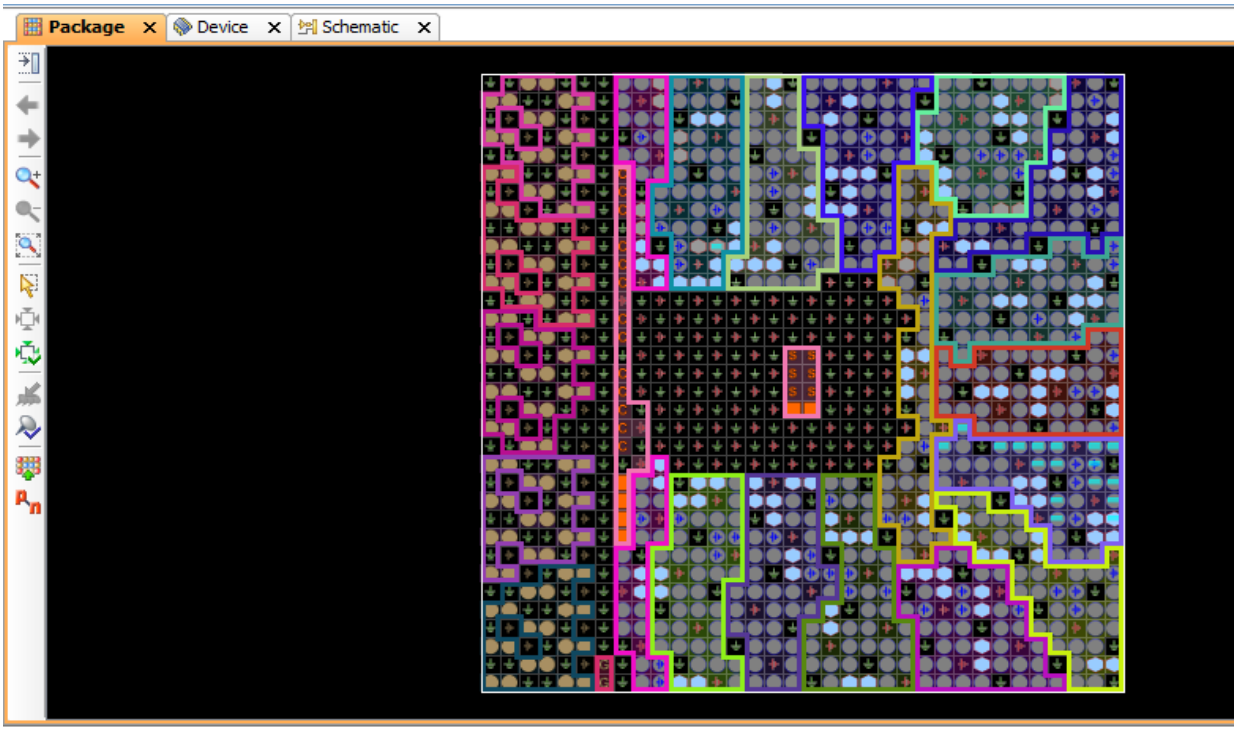

Figure 9. IO Planning of SubByte Transformation

Table 1. Implementation results:

FPGA Device	Virtex6
Maximum combinational path delay	1.436ns
Total equivalent gate count for design	1150
Number of occupied slices	32 / 150720 0.1%
Static Power (mW)	3421.54

V. CONCLUSION

The proposed S-Box gives another option for hardware implementation other than composite field to represent Sub byte transformation. It reduces the complexities of hardware by avoiding the use of multiplicative inverse in Galois field. As compared to the LUT and composite field, the S-Box resulted in smaller area with medium delay. Further work will be carried out to apply the proposed design using ASIC implementation. Optimized and Synthesizable VHDL code is developed for the implementation of encryption process. Therefore, AES can indeed be implemented with reasonable efficiency on an FPGA

REFERENCES

[1] J. Daemen and V. Rijmen, *The Design of Rijndael*: Springer- Verlag New York, Inc., 2002.

[2] S. Tillich, M. Feldhofer, T. Popp, J. Gro, "*Area, delay, and power characteristics of standard-cell implementations of the AES S-Box*," *J. Signal Process. Syst.*, vol. 50, pp. 251-261, 2008.

[3] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "*A Compact Rijndael Hardware Architecture with S-Box Optimization,"* in *Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security:Advances in Cryptology*: Springer-Verlag, 2001.

[4] N. Sklavos and O. Koufopavlou, "*Architectures and VLSI implementations of the AES-Proposal Rijndael*," *Computers, IEEE Transactions on*, vol. 51, pp. 1454-1459, 2002.

[5] Z. Xinmiao and K. K. Parhi, "*High-speed VLSI architectures for the AES algorithm*," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 12, pp. 957-967, 2004.

[6] R. R. Rachh and P. V. Ananda Mohan, "*Implementation of AES S-Boxes using combinational logic*," in *Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on*, 2008, pp. 3294-3297.

[7] C. Nalini, P. V. Anandmohan, D. V. Poomaiah, and V. D. Kulkarni, "*Compact Designs of SubBytes and MixColumn for AES*," in *Advance Computing Conference, 2009. IACC 2009. IEEE International*, 2009, pp. 1241-1247.

[8] Tilborg, Henk C. A. van. "*Fundamentals of Cryptology: A Professional Reference and Interactive Tutorial*", New York Kluwer Academic Publishers, 2002

[9] Peter J. Ashenden, "*The Designer's Guide to VHDL*", 2nd Edition, San Francisco, CA, Morgan Kaufmann, 2002