Low Latency VLSI Architecture of S-box for AES Encryption

Saurabh Kumar[1], V.K. Sharma[2] and K. K. Mahapatra[3]
Department of Electronics & Communication Engineering
National Institute of Technology, Rourkela, India-769008
[1]saurabhsit098@gmail.com, [2]vijay4247@gmail.com, [3]kmaha2@gmail.com

***Abstract*—This paper presents delay improved VLSI architecture of S-box for Advance Encryption Standard (AES) algorithm. The proposed architecture is implemented in FPGA. The delay, area and power comparison with some existing S-box architecture have been done. The comparison results show delay improvement along with low power consumption with constant area in terms of FPGA slices. The silicon validity is done by programming the XC2VP30 device of Xilinx FPGA with VHDL code for the proposed architecture. The architecture is also implemented in ASIC using 0.18 µm standard cell technology library which shows delay improvement of about 16 percent.**

***Keywords*—AES, S-box, Low Latency Design, Composite Field Arithmetic.**

I. INTRODUCTION

Information has become one of the important assets in growing demand of need to store every single important event in everyday life. Messages need to be secured from unauthorized party. Encipherment is one of the security mechanisms to protect information from public access [1]. Encryption hides the original content of a message so as to make it unreadable for anyone, except the person who has the special knowledge to read it [2]. It is done to ensure that private communication is maintained between two users or among a group of users. There are mainly two types of encryption algorithms, private key (also called symmetric key) and public key. Private key algorithms involve only one key, both for encryption and decryption whereas, public key algorithms involve two keys, one for encryption and other for decryption [3]. The public key algorithm is complex and has very high computation time. Private key algorithm is simple and therefore, it is more suitable for faster implementation. National Institute of Standards and Technology (NIST) adopted Advanced Encryption Standard (AES) as the standard for encryption and decryption of blocks of data [4]. The draft is published under the name as FIPS-197 (Federal Information Processing Standard number 197) [5]. AES is a symmetric key block cipher. It encrypts data of block size 128 bits. The AES algorithm is used in diverse application fields like WWW servers, automated teller machines (ATMs), cellular phones and digital video recorders [6]. AES can be implemented in software or hardware. But, hardware implementation is more suitable for high speed applications in real time [7, 8].

The AES is an iterative algorithm and uses four operations in different rounds, namely SubBytes, ShiftRows, MixColumns and KeyAdditions transformations [9]. SubBytes transformation(also called substitution) is a non-linear operation in AES wherein each byte of a state is mapped to a different value. The SubBytes transformation is done through S-box. There are two techniques to perform substitutions, (i) using ROM table [10−12], and (ii) using composite field arithmetic [13−15]. Substitution is the most complex steps in terms of cost and implementation [13]. Therefore, its hardware optimization for VLSI implementation is very important to reduce the area and power of the AES architecture. ROM based approach requires high amount of memory and also it causes low latency because of ROM access time. Therefore, composite field arithmetic is more suitable for S-box implementation.

In this paper, we propose delay improved VLSI architecture for S-box. The FPGA implementation of the architecture is done along with comparison with some existing transformation techniques. The proposed architecture has delay improvement and low power consumption. The silicon validation of the architecture is done by programming XC2VP30 device on Virtex-II Pro FPGA board. The proposed architecture is also implemented in ASIC using 0.18 µm standard cell technology library.

The remaining part of this paper is as follows. Section II describes AES algorithm and composite field arithmetic for the realization of S-box. Proposed structure of S-box is described in section III. Section IV contains the FPGA/ASIC implementation results along with comparison with some existing S-box structures. Conclusion is drawn in section V.

II. The AES Encryption Algorithm and S-box Realization in Composite Field

The AES algorithm encrypts data in blocks of 128 bits. It uses three key sizes, 128 bits, 192 bits and 256 bits in three versions. AES uses three different round operations. TABLE I shows the number of rounds in three different versions of AES. But, in each version final round key is 128 bits. The initialization is done by adding first round key (128 bits) with 128 bits plaintext. In subsequent steps, the following transformations are done: SubBytes, ShiftRows, MixColumns and AddRoundKey. The last round is different from the previous rounds as there is no MixColumns transformation. Figure 1 shows the rounds in AES. The internal 128 bits data in AES are represented in the form of 4x4 square matrix containing elements of size 8 bits and named as state elements. The SubBytes transformations involve mapping each state element with another data. The relation between input and output of this transformation is non-linear. The SubBytes transformation done through S-box mapping is computationally efficient when implemented using ROM. But, it is not efficient for applications requiring very high throughput as ROM accessing involves one complete clock cycle for mapping one 8-bits state element, and consequently 16 clock cycles are required to transform 128 bits of data (16 bytes). To increase the throughput, parallel ROMs are required resulting in large size of chip area. Therefore, a more feasible solution to implement S-box is by using composite field arithmetic which uses only logic elements in the implementation.

TABLE I: Round Key Size and Number of Rounds in AES

Cipher Key size	No. of Rounds (Nr)	Round Key size
128 bits	10	128 bits
192 bits	12	128 bits
256 bits	14	128 bits

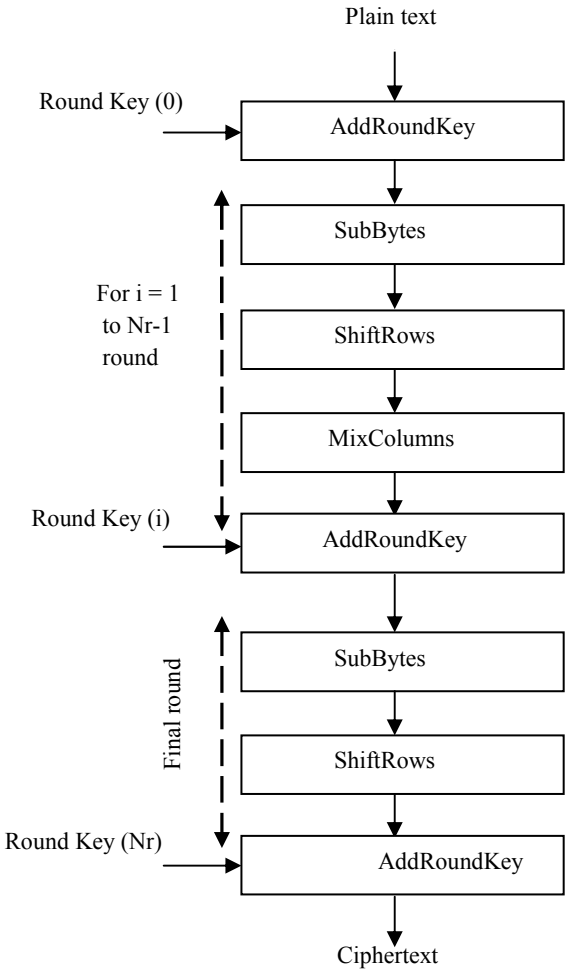

Figure 1. Rounds in AES encryption algorithm

Figure 2 shows steps for the one byte forward and inverse transformation using composite field arithmetic [15]. One major operation involves here is finding the multiplicative inverse in GF (2^8). This can be done by breaking the GF (2^8) elements in GF (2^4). i.e., any arbitrary polynomial in GF (2^8) can be represented as bx+c using an irreducible polynomial x^2+Ax+B. Here, b is the most significant nibble and c is the least significant nibble. The multiplicative inverse can be found by using the following expression [16].

$$
\begin{aligned}
&(bx+c)^{-1} \\
&= b(b^2B+bcA+c^2)^{-1}x+(c+bA)(b^2B+bcA+c^2)^{-1} \\
&= b(b^2\lambda+c(b+c))^{-1}x+(c+b)(b^2\lambda+c(b+c))^{-1}
\end{aligned}
\tag{1}
$$

where, A=1, B=λ, as the irreducible polynomial used is x^2+x+λ. Figure 3 shows the block diagram to find the multiplicative inverse in GF(2^8) using GF(2^4) [16]. Figure 4 shows the meanings of symbols used in Figure 3. The mapping structure in different fields along with the irreducible polynomials is given by Eq. (2).

$$
\left.
\begin{aligned}
&GF(2^2) \longrightarrow GF(2) && : x^2+x+1 \\
&GF((2^2)^2) \longrightarrow GF(2^2) && : x^2+x+\varphi \\
&GF(((2^2)^2)^2) \longrightarrow GF((2^2)^2) && : x^2+x+\lambda
\end{aligned}
\right\}
\tag{2}
$$

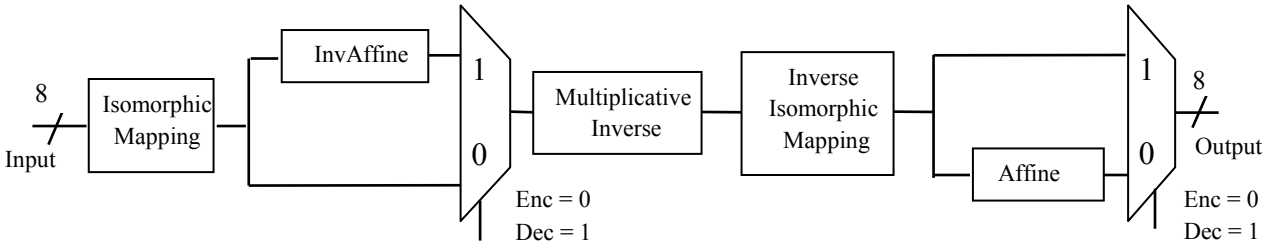

Figure 2. SubBytes and Inverse SubBytes transformation in composite field [15]

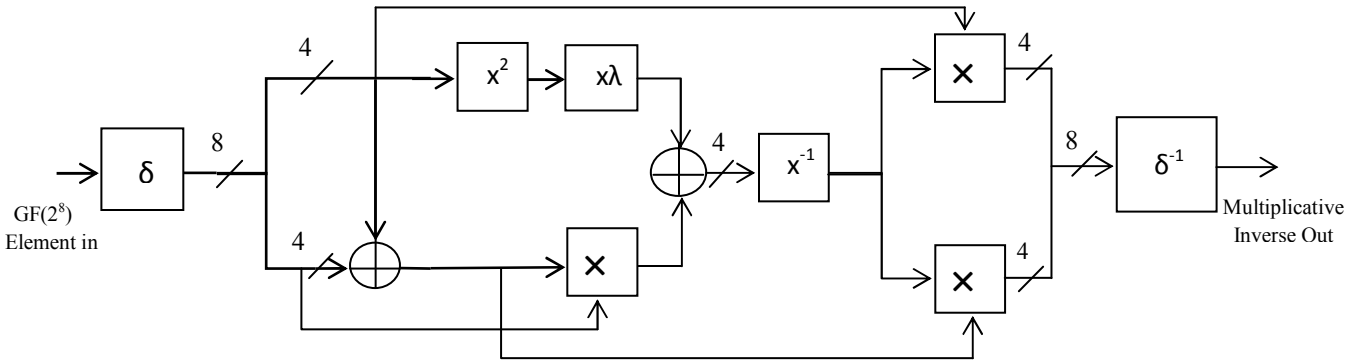

Figure 3.Block diagram to find the Multiplicatiive Inverse for the S-box [16]

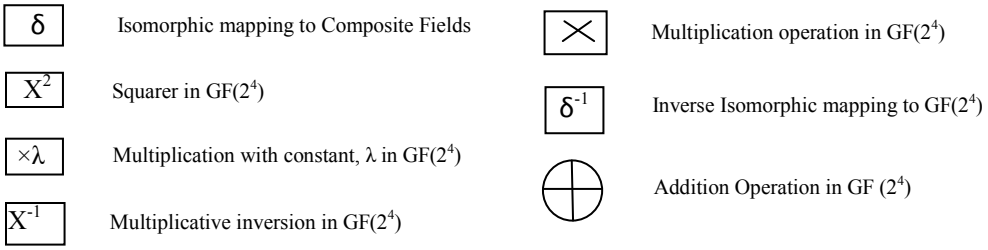

Figure 4.Meaning of symbols used in Figure 3

III. Proposed Multiplicative Inverse for S-box

The expression $b^2\lambda+c(b+c)$ in Eq. (1) can be written as,

$$b^2\lambda + bc + c^2 = b(b\lambda + c) + c^2 \tag{3}$$

Representing b, c and λ as,

$$b= b_Hx+b_L,\ c= c_Hx+c_L,\ \lambda= \lambda_Hx+ \lambda_L$$

where b_H and b_L are the upper and lower 2-bits of b.

Similarly, c_H and c_L are the upper and lower 2-bits of c and λ_H and λ_L are the upper and lower 2-bits of λ. bλ, i.e., multiplication with λ can be written as,

$$b\lambda = b_H\lambda_H x^2 + b_L\lambda_H x \tag{4}$$

and therefore,

$$b\lambda + c = b_H\lambda_H x^2 + b_L\lambda_H x + c_H x + c_L \tag{5}$$

Now from Eq. (5),

$$
\begin{aligned}
& b(b\lambda + c) \\
& = (b_H \lambda_H x^2 + b_H \lambda_H + c_H x + c_L) \times (b_H x + b_L) \\
& = (b_H{}^2 \lambda_H \varphi + b_H{}^2 \lambda_H + c_H b_H + c_L b_H + b_L{}^2 \lambda_H + b_L c_H) x \\
& + (b_L c_L + b_H{}^2 \lambda_H \varphi + b_H c_H \varphi)
\end{aligned} \tag{6}
$$

Here, $\lambda = (1100)_2$ and $\varphi = (10)_2$. Performing operations in GF($(2^2)^2$), the following values can be obtained in terms of upper (b) and lower nibble (c) bits.

$$
\begin{aligned}
& op(0) \\
& = b(0)c(0) + b(1)c(1) + b(2)c(3) + b(3)c(3) \\
& + b(3)c(2) + in(0) + in(3) + in(5) + in(7) \\
& op(1) \\
& = b(0)c(1) + b(1)c(0) + b(1)c(1) + b(2)c(2) \\
& + b(2)c(3) + b(3)c(2) + in(2) + in(3) + in(5) + in(6) \\
& op(2) \\
& = b(0)c(2) + b(1)c(3) + b(2)c(0) + b(2)c(2) \\
& + b(3)c(1) + b(3)c(3) + in(1) + in(2) + in(4) + in(6) \\
& op(3) \\
& = b(3)(in(0) + in(3) + in(6)) + c(3)(b(0) + b(1) + b(2)) \\
& + b(2)c(1) + b(1)c(2) + in(1) + in(3) + in(4)
\end{aligned} \tag{7}
$$

where, $in(0)$, $in(1)$, $in(2)$, $in(3)$, $in(4)$, $in(5)$, $in(6)$ are the input bits in the isomorphic mapping module (δ).

Figure 5 shows the proposed structure for finding the multiplicative inverse in GF (2^8) where *op* is as obtained from the Eq. (7).

IV. FPGA/ASIC Implementation of S-box Using Proposed Structure for Multiplicative Inverse

The S-box in AES has been implemented in Xilinx FPGA using proposed multiplicative structure of Figure 5. The device taken for the implementation is XC2VP30 on Virtex-II Pro board. For the comparison, conventional S-box architecture in [16] has also been implemented using multiplicative inverse structure of Figure 3 in the XC2VP30 device. TABLE II shows the hardware utilization summary in terms of Slices and LUTs. Power consumption has been measured by Xpower analyser tool in ISE 10.1 design suite. From TABLE II, it is evident that there is improvement in delay and power consumption in proposed structure as compared to the conventional S-box structure [16]. Figure 6 shows the sample hardware output obtained from FPGA through ChipScope Pro logic analyser after programming

TABLE II: Round Key Size and Number of Rounds in Three Versions of AES

	Proposed structure	Conventional Structure in [16]	Structure in [14]
Device	XC2VP30	XC2VP30	XC2V1000
# of Slices	40	41	153
# of 4-input LUTs	71	72	NA
Max. Delay (ns)	15.0	15.6	10.82
Total Dynamic Power (W)	7.271	9.74	NA

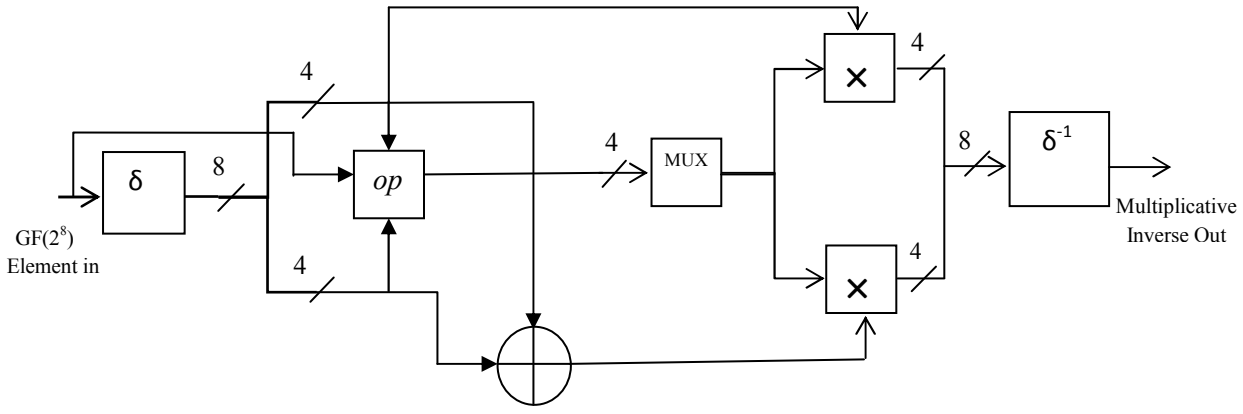

Figure 5.The proposed multiplicative inverse structure in GF (2^8)

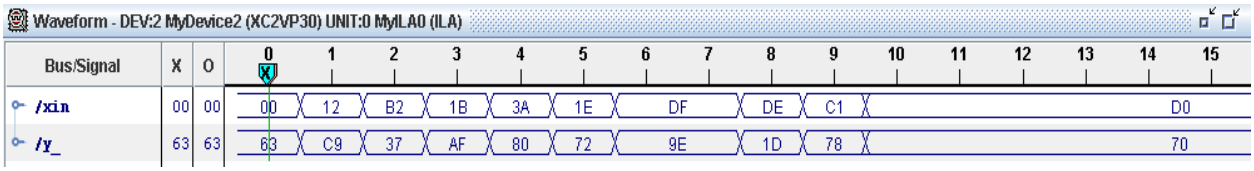

Figure 6. Hardware implementation Result of S-box obtained from XC2VP30 device using ChipScope pro logic analyser

the XC2VP30 device. TABLE III shows the delay, power and area comparison in ASIC implementation using 0.18 µm standard cell technology library. The proposed architecture has delay improvement of about 0.9 ns (=16 %) as comapared to the conventional S-box architecture with little area cost. Throughput per unit area is an important criterion to measure the efficiency of the hardware [17]. TABLE III confims that the proposed S-box architecture has high hardware efficiency.

TABLE III: ASIC IMPLEMENTATION AND COMPARISON OF AES ARCHITECTURE

Technology 0.18 µm	Proposed structure	Conventional Structure in [16]
Area (μm^2)	3968	3715
Gate Counts	404.89	379.08
Delay	4.6 (Improvement =16 % from conventional)	5.51
Total Dynamic Power (mW)	2.4985	2.4380
Efficiency (kbps/μm^2) (Throughput/Area)	438.25	390.57

V. CONCLUSION

We have proposed a delay improved VLSI architecture of S-box for AES algorithm. The proposed architecture is implemented in Xilinx FPGA as well as 0.18 µm ASIC technology. It shows delay improvement of about 16 percent as compared to conventional S-box structure in ASIC. There is delay improvement of 0.6 ns in FPGA implementation as well. Silicon validation is done by programming FPGA with VHDL code of the proposed structure.

ACKNOWLEDGMENT

The authors acknowledge to DIT (Ministry of Information & Communication Technology) for the financial support for carrying out this research work.

REFERENCES

[1] B.A. Forouzan and D. Mukhopadhyay, Cryptography and Network Security, 2nd Ed., Tata McGraw Hill, New Delhi, 2012.

[2] M. I. Soliman, G. Y. Abozaid, "FPGA implementation and performance evaluation of a high throughput crypto coprocessor," *Journal of Parallel and Distributed Computing*, Vol. 71 (8), pp.1075-1084, Aug. 2011.

[3] V. K. Pachghare, Cryptography and information security, E. E. Ed., PHI Learning, New Delhi, 2009.

[4] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "A Lightweight High-Performance Fault Detection Scheme for the Advanced Encryption Standard Using Composite Fields," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 19 (1), pp. 85-91, Jan. 2011.

[5] Federal Information Processing Standards Publication 197 (FIPS 197), available online, http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[6] X. Zhang, K. K. Parhi, "High-Speed VLSI Architectures for the AES Algorithm," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 12 (9), pp. 957-967, Sep. 2004.

[7] M. Jridi and A. AlFalou, "A VLSI implementation of a new simultaneous images compression and encryption method," *2010 IEEE International Conference on Imaging Systems and Techniques (IST),* pp.75-79, July 2010.

[8] Chih-Pin Su, Tsung-Fu Lin, Chih-Tsun Huang, and Cheng-Wen Wu, "A High-Throughput Low-Cost AES Processor," *IEEE Communications Magazine*, Vol.41 (12), pp.86-91, Dec. 2003.

[9] L. Ali, I. Aris, F. S. Hossain and N. Roy, "Design of an ultra high speed AES processor for next generation IT security," *Computers and Electrical Engineering*, Vol.37 (6), pp.1160-1170, Nov. 2011.

[10] K.H. Chang, Y.C. Chen, C. C. Hsieh, C. W. Huang and C. J. Chang, "Embedded a Low Area 32-bit AES for Image Encryption/Decryption Application," *IEEE International Symposium on Circuits and Systems*, pp. 1922-1925, May 2009.

[11] J. M. G. Criado, M. A. V. Rodriguez, J. M. S. Perez, J. A. G. Pulido, "A new methodology to implement the AES algorithm using partial and dynamic reconfiguration," *Integration, the VLSI Journal*, Vol.43(1), pp. 72-80, Jan. 2010.

[12] J. V. Dyken, J. G. Delgado-Frias, "FPGA schemes for minimizing the power-throughput trade-off in executing the Advanced Encryption Standard algorithm," *Journal of Systems Architecture*, Vol.56(2–3), pp. 116-123, Mar. 2010.

[13] I. Hammad, K. E. Sankary and E. E. Masry, "High-Speed AES Encryptor With Efficient Merging Techniques," *IEEE Embedded Systems Letters*, Vol.2 (3), pp.67-71, Sept. 2010.

[14] N. Ahmad, R. Hasan, W. M. Jubadi, "Design of AES S-Box using combinational logic optimization," *IEEE Symposium on Industrial Electronics & Applications*, pp.696-699, Oct. 2010.

[15] N. Ahmad, S. M. R. Hasan, "Low-power compact composite field AES S-Box/Inv S-Box design in 65 nm CMOS using Novel XOR Gate," *Integration, the VLSI Journal*, Article in Press.

[16] Edwin NC Mui, "Practical Implementation of Rijndael S-Box Using Combinational Logic", Custom R&D Engineer Texco Enterprise Pvt.Ltd.

[17] Y.H. Chen, T.Y. Chang, and C.Yi Li, "High Throughput DA-Based DCT With High Accuracy Error-Compensated Adder Tree," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 19 (4), pp. 709-714, Apr. 2011.