An Improved VLSI Architecture of S-box for AES Encryption

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

***Abstract*—This paper presents an improved VLSI architecture of S-box for AES encryption system. Certain basic blocks in conventional architecture are replaced by efficient multiplexers and an optimized combinational logic to facilitate speed improvement. The proposed as well as conventional architecture are implemented in Xilinx FPGA and 0.18 µm standard cell ASIC technology. ASIC implementation indicates speed enhancement while maintaining constant area compared to conventional architecture. FPGA implementation also confirms speed improvement of about 0.6 ns along with low utilization of FPGA fabrics. Furthermore, there is significant power improvement (155 %) compared to conventional structure.**

***Keywords-* S-box, Composite field arithmetic, AES encryption, FPGA implementation.**

I. Introduction

National Institute of Standards and Technology (NIST) adopted Advanced Encryption Standard (AES) as the standard for block data encipherement [1]. AES is a symmetric key cipher that encrypts data blocks of three different sizes, namely 128 bits, 192 bits and 256 bits with round keys 128 bits, 192 bits and 256 bits in its three different versions [2, 3]. However, all three versions use 128 bits round key, which is generated by round key generation process, for encryption/decryption [4]. SubBytes, ShiftRows, MixColumns and AddRound keys are the transformations steps involved in AES encryption algorithm [5]. SubBytes transformation (also called byte substitution) is the most computing step in which each 8-bits element of a state matrix is transformed to have a new value [6, 7]. SubBytes transformation is performed by using a substitution table, called S-box [8].

The SubBytes transformation using table is implemented in VLSI using ROM which stores the table values [3, 6, 9]. However, ROM based design limits the performance because of ROM access time. Therefore, alternate implementation of S-box without ROM is performed using combinational logic which has high speed of data encryption [10−13].

Speed improvement along with area reduction has been the most challenging research in VLSI implementation. In this paper, we have further optimized delay and area of the conventional S-box architecture by using some efficient logic and multiplexers in the critical path. The proposed optimized S-box architecture has been implemented in 0.18 µm ASIC technology as well as Xilinx FPGA. The proposed architecture of S-box shows delay and area improvement with respect to conventional S-box architecture. Also, there is significant power improvement in FPGA implementation in proposed method.

The remaining part of the paper is organized as follows. Section II describes combinational S-box architecture using combinational logic in brief. In section III, two optimization techniques for delay improvement in critical path of conventional S-box architecture have been explained. Section IV contains the VLSI/FPGA implementation results of both conventional as well as proposed design along with delay, area and power comparisons. Conclusions are drawn in section V.

II. S-box Realization in Composite Field

Figure 1 shows the conventional S-box architecture using composite field arithmetic. The meaning of the symbol used in this architecture has been shown in Figure 2. For the S-box mapping, following are the steps. Isomorphic mapping is the first step performed on the 8 bits sub byte input. Output of the isomorphic mapping is given to input of multiplicative inverse (MI) module. Subsequently, inverse isomorphic mapping and affine transformations are the steps that follow. Detail explanation can be found in [13]. MI in GF (2^4) represented by the symbol x^{-1} and multiplication in GF (2^4) are the two main components falls in the critical path of the design. MI in GF (2^4) consists of complex logic given by [12],

$$\left.\begin{aligned}
q_3{}^{-1} &= q_3 + q_3q_2q_1 + q_3q_0 + q_2 \\
q_2{}^{-1} &= q_3q_2q_1 + q_3q_2q_0 + q_3q_0 + q_2 + q_2q_1 \\
q_1{}^{-1} &= q_3 + q_3q_2q_1 + q_3q_1q_0 + q_2q_0 + q_2 + q_1 \\
q_0{}^{-1} &= q_3q_2q_1 + q_3q_2q_0 + q_3q_1 + q_3q_1q_0 + q_3q_0 \\
&\quad + q_2 + q_2q_1 + q_2q_1q_0 + q_1 + q_0
\end{aligned}\right\} \tag{1}$$

where, $q_3{}^{-1}q_2{}^{-1}q_1{}^{-1}q_0{}^{-1}$ is 4-bits MI of 4-bit value $q_3q_2q_1q_0$ and + sign indicates XOR operation.

The complex operation multiplication in GF (2^4) is realized by the logic as shown in Figure 3(a). This logic structure uses multiplication in GF (2^2) as the basic component as shown in Figure 3(b).

It is obvious that delay optimization of the two components mentioned above can lead to the overall delay improvement of the S-box architecture.

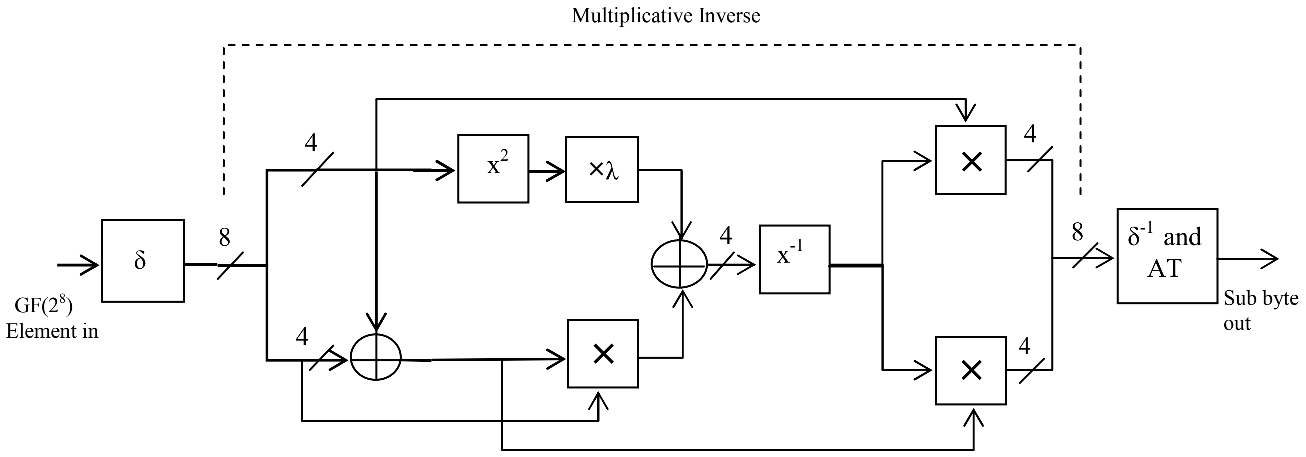


Figure 1. A conventional S-box architecture in composite field [7]

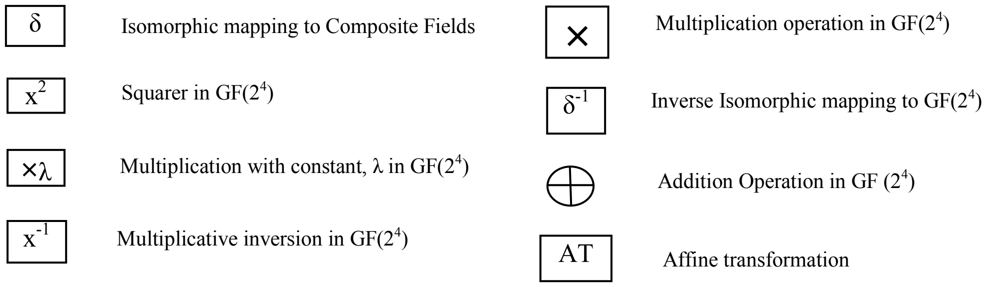
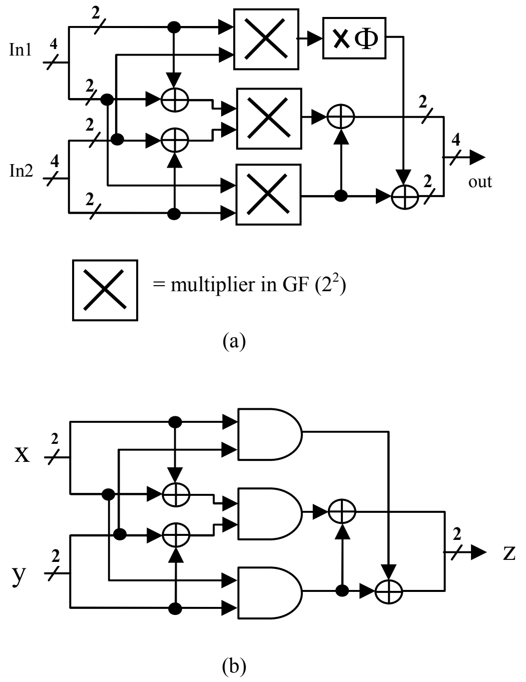

δ
Isomorphic mapping to Composite Fields
x^2
Squarer in GF(2^4)
×λ
Multiplication with constant, λ in GF(2^4)
x^-1
Multiplicative inversion in GF(2^4)
Multiplication operation in GF(2^4)
δ^-1
Inverse Isomorphic mapping to GF(2^4)
Addition Operation in GF (2^4)
AT
Affine transformation

Figure 2.Meaning of symbols used in Figure 1



(a)

(b)

Figure 3. Circuit for (a) multiplier in GF (2^4) and (b) multiplier in GF (2^2)

III. The Proposed Logic Circuits for Delay and Area Improvement

We have optimized the delay and area of the MI in GF (2^4) as well as multiplication in GF (2^2).

A. Proposed logic for MI in GF (2^4)

From Eq. (1), it is evident that the realization of MI in GF (2^4) requires a number of exclusive-or (XOR) gates. By eliminating the XOR gates, delay and area can be reduced. TABLE I shows the input and output combination of MI in GF (2^4). The input combinations can be divided into two equal halves. In the first half, MSB will have value '0' and in the second half, MSB will be '1'. This can be realized by a multiplexer, wherein, for '0' MSB in input, the 4-bit output will be given by combination of three input bits (except MSB). Similarly, for MSB= '1', the 4-bit output will have 3-bits input combination (except MSB). The combinational logic for first half in terms of three input bits is given by the Eq. (2a). Eq. (2b) represents the combinational logic for the second half. By using a multiplexer, one of the outputs, either from (2a) or from (2b) can be selected depending on the MSB of the input. It is obvious from Eq. (2a) and Eq. (2b) that the combinational logic contains only OR and AND gates instead of XOR gates used as in Eq. (1).

TABLE I. MI IN GF (2^4)

Input to MI in GF (2^4)	Output from MI in GF (2^4)	
$q_3q_2q_1q_0$	$q_3^{-1}q_2^{-1}q_1^{-1}q_0^{-1}$	
0000	0000	First half
0001	0001	
0010	0011	
0011	0010	
0100	1111	
0101	1100	
0110	1001	
0111	1011	
1000	1010	Second half
1001	0110	
1010	1000	
1011	0111	
1100	0101	
1101	1110	
1110	1101	
1111	0100	

$$\left.\begin{aligned}
q_3^{-1} &= \left(q_2\overline{q_1}\,\overline{q_0}\right) or \left(q_2\overline{q_1}q_0\right) or \left(q_2q_1\overline{q_0}\right) or \left(q_2q_1q_0\right)\\
q_2^{-1} &= \left(q_2\overline{q_1}\,\overline{q_0}\right) or \left(q_2\overline{q_1}q_0\right)\\
q_1^{-1} &= \left(\overline{q_2}q_1\overline{q_0}\right) or \left(\overline{q_2}q_1q_0\right) or \left(q_2\overline{q_1}\,\overline{q_0}\right) or \left(q_2q_1q_0\right)\\
q_0^{-1} &= \left(\overline{q_2}\,\overline{q_1}q_0\right) or \left(\overline{q_2}q_1\overline{q_0}\right) or \left(q_2\overline{q_1}\,\overline{q_0}\right) or \left(q_2q_1\overline{q_0}\right)\\
&\quad or\left(q_2q_1q_0\right)
\end{aligned}\right\} \tag{2a}$$

$$\left.\begin{aligned}
q_3^{-1} &= \left(\overline{q_2}\,\overline{q_1}\,\overline{q_0}\right) or \left(\overline{q_2}q_1\overline{q_0}\right) or \left(q_2\overline{q_1}q_0\right) or \left(q_2q_1\overline{q_0}\right)\\
q_2^{-1} &= \left(\overline{q_2}\,\overline{q_1}q_0\right) or \left(\overline{q_2}q_1q_0\right) or \left(q_2\overline{q_1}\,\overline{q_0}\right) or \left(q_2\overline{q_1}q_0\right)\\
&\quad or\left(q_2q_1\overline{q_0}\right) or \left(q_2q_1q_0\right)\\
q_1^{-1} &= \left(\overline{q_2}\,\overline{q_1}\,\overline{q_0}\right) or \left(\overline{q_2}\,\overline{q_1}q_0\right) or \left(\overline{q_2}q_1q_0\right) or \left(q_2\overline{q_1}q_0\right)\\
q_0^{-1} &= \left(\overline{q_2}q_1q_0\right) or \left(q_2\overline{q_1}\,\overline{q_0}\right) or \left(q_2q_1\overline{q_0}\right)
\end{aligned}\right\} \tag{2b}$$

B. Proposed logic for multiplication in GF (2^2)

From Figure 3(b), multiplication in GF (2^2) can be expressed as,

$$\left.\begin{aligned}
z(0) &= x(1)y(1) \oplus x(0)y(0)\\
z(1) &= x(1)y(1) \oplus x(0)y(1) \oplus x(1)y(0)
\end{aligned}\right\} \tag{3}$$

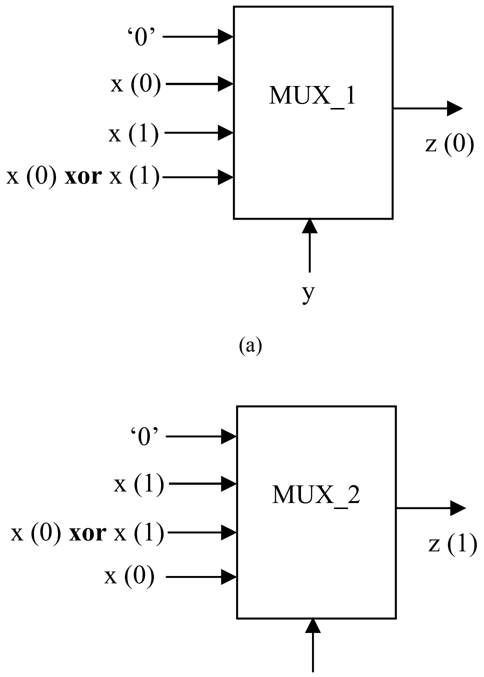
Eq. (3) can be implemented using two 4:2 parallel multiplexers as follows. Suppose y is the select line. Then, for different values of y, multiplication result z, from Eq. (3), will have values as given in TABLE II. Figure 4 shows the two multiplexers. From the Figure 3(b), it is evident that there are two XOR gates and one AND gate in the critical path of GF (2^2) multiplication. One XOR gate has been eliminated from the critical path by using 4:2 multiplexer, i.e., there is one XOR gate and one multiplexer only in critical path, as compared to two XOR gates in Figure 3(b).

IV. VLSI/FPGA IMPLEMENTATION RESULTS AND COMPARISONS

The proposed two techniques, one for MI in GF (2^4) and other one multiplication in GF (2^2) have been implemented separately in ASIC using 0.18 µm technology. TABLE III shows the comparison results of MI with conventional. TABLE IV shows the comparison results of multiplication in GF (2^2). It is evident that both the proposed methods are delay and area efficient. The complete S-box has been implemented and TABLE V shows the ASIC implementation results and TABLE VI shows the FPGA implementation results along with comparison with conventional S-box architecture. The proposed method has delay improvement of about 0.2 ns in ASIC and about 0.6 ns in FPGA. There is significant improvement in power consumption in FPGA along with area improvement in terms of FPGA slices. Figure 5 shows the simulation output of S-box implementation in Xilinx ISE 10.1.

TABLE II. RESULTS OF MULTIPLICATION IN GF(2^2)

Value of y	z(0)	z(1)
00	‘0’	‘0’
01	x(0)	x(1)
10	x(1)	x(0) **xor** x(1)
11	x(0) **xor** x(1)	x(0)



(a)

(b)

Figure 4. 4:2 multiplexer for (a) LSB output and (b) MSB output for 2 bits output of multiplication in GF (2^2)

TABLE III. COMPARISON OF MI IN GF (2^4) IN ASIC

Technology 0.18 µm	Conventional structure	Proposed structure
Area (µm^2)	352	279.41
Total Dynamic Power (µW)	97.58	62.93
Delay (ns)	0.79	0.52

TABLE IV. COMPARISON OF MULTIPLICATION IN GF (2^2) IN ASIC

Technology 0.18 µm	Conventional structure	Proposed structure
Area (µm^2)	123.00	126.40
Total Dynamic Power (µW)	28.64	24.45
Delay (ns)	0.41	0.23

TABLE V. COMPARISON OF S-BOX IMPLEMENTATION IN ASIC

Technology 0.18 µm	Conventional structure	Proposed structure
Area (µm^2)	3715	3702
Total Dynamic Power (mW)	2.44	2.50
Delay (ns)	5.51	5.31

TABLE VI. FPGA IMPLEMENTATION RESULTS AND COMPARISONS

	Conventional structure	Structure in [11]	Proposed structure
Device	**XC2VP30**	**XC2V1000**	**XC2VP30**
# of Slices	41	153	37
# of 4-input LUTs	72	NA	66
Max. Delay (ns)	15.6	10.82	15.0
Total Dynamic Power (W)	9.74	NA	3.84 (at Max. clock frequency)

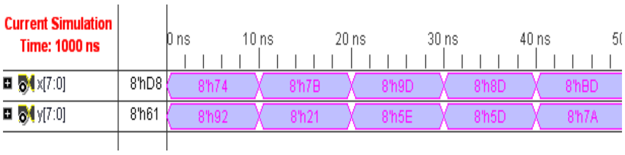


Figure 5. VHDL simulation of S-box in Xilinx ISE 10.1

V. CONCLUSIONS

An optimized architecture of S-box for AES encryption is proposed in this paper. This novel architecture is implemented both in ASIC as well as FPGA. The ASIC implementation indicates speed improvement compared to conventional structure while maintaining area constant. FPGA implementation shows improvement in delay and area while a significant enhancement in terms of power compared to conventional architecture.

REFERENCES

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

[2] 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.

[3] 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.

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

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

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