IJDACR
ISSN: 2319-4863

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research
Website: www.ijdacr.com (Volume 1, Issue 12, July 2013)

FPGA Implementation of Improved S-BOX Architecture for Advanced Encryption Standard

Prahlad Kumar Khandekar
prahladkh@gmail.com
M.Tech. Scholar, ET&T Department
Chouksey Engg. College Bilaspur

Sachin Meshram
sachinm288@gmail.com
Asst. Professor, ET&T Department
Chouksey Engg. College Bilaspur

Abstract — **Advance Encryption Standard (AES) is one of the most popular cryptographic algorithm now a days providing integrity, authentication and security. The Substitution block, used for security better known as S-BOX is the key element of Advance Encryption Standard algorithm. Different algorithm presented in previous work which are lagging behind in few parameters, which is corrected and implemented in this paper. Proposed architecture is implemented in VHDL Using Xilinx ISE 12.1 on device xc3s1200e-5fg320 of Spartan family.**
Keywords — **S-Box, AES, Galois Field, VHDL, Spartan.**

I. Introduction

AES encryption is an efficient scheme for both hardware and software implementation. Much work has been presented on hardware implementations of AES using field programmable gate arrays, and comprehensive analyses of the performance of the AES finalists was presented based on FPGA implementations, before Rijndael was selected as the AES algorithm. One of the most common and straight forward implementation of the S-Box for the SubByte operation which was done in previous work was to have the pre-computed values stored in a ROM based lookup table. In this implementation, all 256 values are stored in a ROM and the input byte would be wired to the ROM's address bus. However, this method suffers from an unbreakable delay since ROMs have a fixed access time for its read and write operation. Furthermore, such implementation is expensive in terms of hardware.

To Speed the operation a pre computation based technique is proposed which is much faster in terms of input to output delay but consumes much power and area on chip.

However the need is to design an S-BOX which is efficient in terms of:

- Speed
- Area
- Power

II. AES Algorithm

The Advanced Encryption Standard (AES) also called the Rijndael algorithm, specifies a FIPS (Federal Information Processing Standards Publications) approved cryptographic algorithm that can be used to protect electronic data. 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; decrypting the cipher-text converts the data back into its original form, called plaintext. The AES algorithm is capable of using cryptographic keys of 128, 192, and 256 bits to encrypt and decrypt data in blocks of 128 bits.

Encryption Process

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

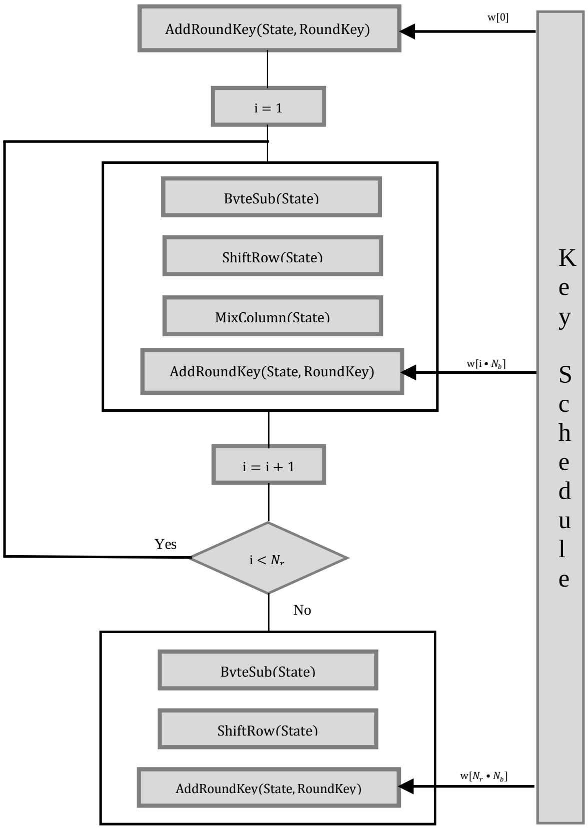

Figure 1: The Encryption process of AES algorithm

IJDACR
ISSN: 2319-4863

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research
Website: www.ijdacr.com (Volume 1, Issue 12, July 2013)

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.

Decryption Process

The Decryption process of Advanced Encryption Standard algorithm is presented below, in figure 2.

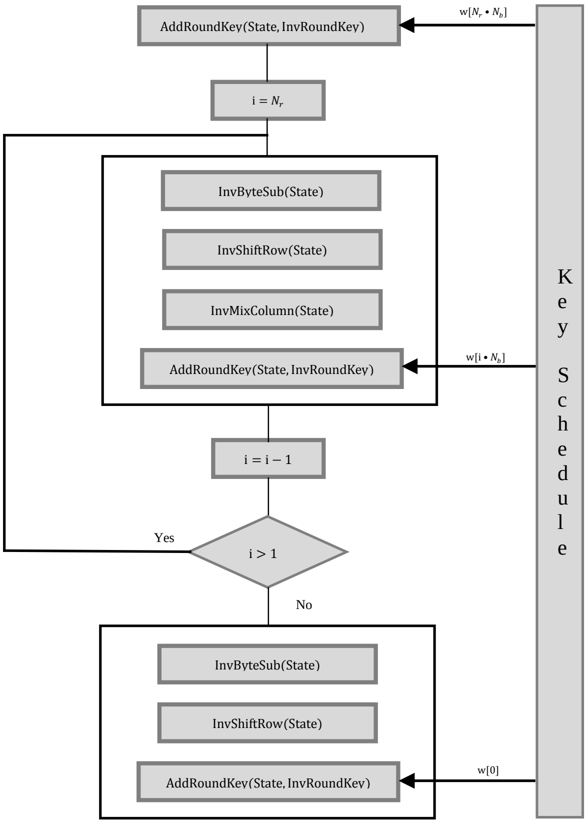

Figure 2: The Decryption process of AES algorithm

This process is direct inverse of the Encryption process. All the transformations applied in Encryption process are inversely applied to this process. Hence the last round values of both the data and key are first round inputs for the Decryption process and follows in decreasing order.

III. Proposed S-Box Design Method

One of the most common and straight forward implementation of the S-Box for the SubByte operation which was done in previous work was to have the pre-computed values stored in a ROM based lookup table. In this implementation, all 256 values are stored in a ROM and the input byte would be wired to the ROM's address bus. However, this method suffers from an unbreakable delay since ROMs have a fixed access time for its read and write operation [7]. Furthermore, such implementation is expensive in terms of hardware. A more refined way of implementing the S-Box is to use combinational logic. Such examples of work that implements the S-Box using this method were [4], [5] and [7]. This S-Box has the advantage of having small area occupancy, in addition to be capable of being pipelined for increased performance in clock frequency. The S-Box architecture discussed in this paper is based on the combinational logic implementation. The steps involved in constructing the multiplicative inverse module for the S-Box using composite field arithmetic is expressed as under. Since both the SubByte and InvSubByte transformation are similar other than their operations which involve the Affine Transformation and its inverse, therefore only the implementation of the SubByte operation will be discussed in this paper. The multiplicative inverse computation will first be covered and the affine transformation will then follow to complete the methodology involved for constructing the S-Box for the SubByte operation [2]. For the InvSubByte operation, we can reuse multiplicative inversion module and combine it with the Inverse Affine Transformation, as shown in Figure 3.

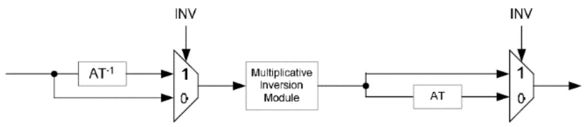

Figure 3: Combined SubByte and InvSubByte sharing a common multiplicative inversion module

The individual bits in a byte representing a GF (2^8) element can be viewed as coefficients to each power term in the GF (2^8) polynomial. For instance, $\{10001011\}_2$ is representing the polynomial $q^7 + q^3 + q + 1$ in GF (2^8). From [3], it is stated that any arbitrary polynomial can be represented as $bx + c$, given an irreducible polynomial of $x^2 + Ax + B$. Thus, element in GF (2^8) may be represented as $bx + c$ where b is the most significant nibble while c is the least significant nibble. From here, the multiplicative inverse can be computed using the equation below [3].

$$(bx + c)^{-1} = b(b^2B + bcA + c^2)^{-1}x + (c + bA)(b^2B + bcA + c^2)^{-1}$$

From [4], the irreducible polynomial that was selected was $x^2 + x + \lambda$. Since $A = 1$ and $B = \lambda$, then the equation could be simplified to the form as shown below.

$$(bx + c)^{-1} = b(b^2\lambda + c(b + c)^{-1}x + (c + b)\,(b^2\lambda + c(b + c)^{-1}$$

IJDACR
ISSN: 2319-4863

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research
Website: www.ijdacr.com (Volume 1, Issue 12, July 2013)

The above equation indicates that there are multiply, addition, squaring and multiplication inversion in GF (2^4) operations in Galois Field. Each of these operators can be transformed into individual blocks when constructing the circuit for computing the multiplicative inverse. From this simplified equation, the multiplicative inverse circuit GF (2^8) can be produced as shown in Figure 4.

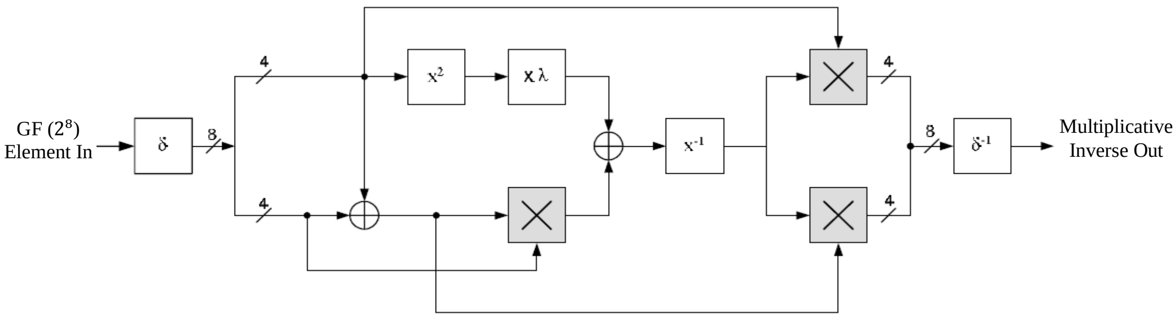

Figure 4: S-Box Architecture

The legends for the blocks within the multiplicative inversion module from above are illustrated in the Figure 5 below

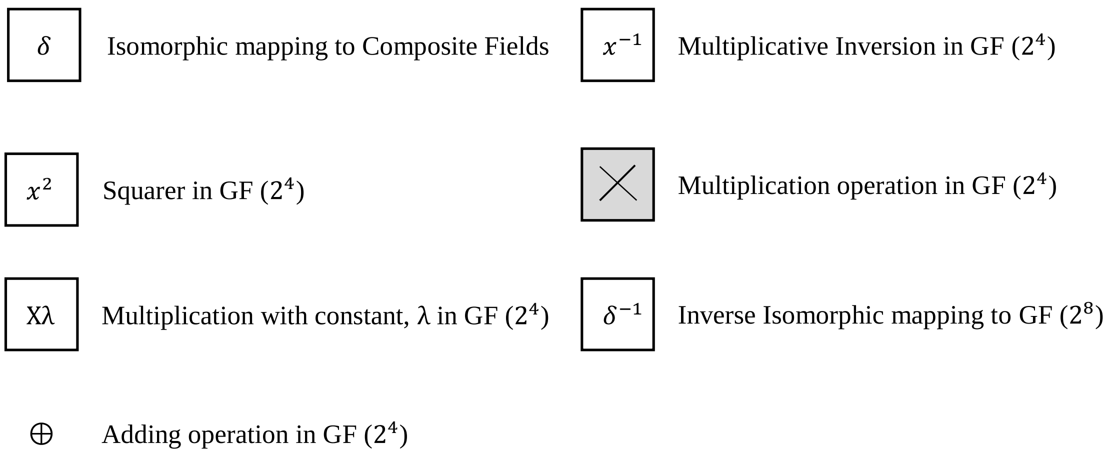

Figure 5: Legends for the building blocks within the multiplicative inversion module

IJDACR
International Journal Of Digital Application & Contemporary Research

International Journal of Digital Application & Contemporary research
Website: www.ijdacr.com (Volume 1, Issue 12, July 2013)

IV. SIMULATION AND RESULTS

Here we show the FPGA implementation and results of given architectures for S-box implemented on Xilinx xc3s1200e-5fg320 device.

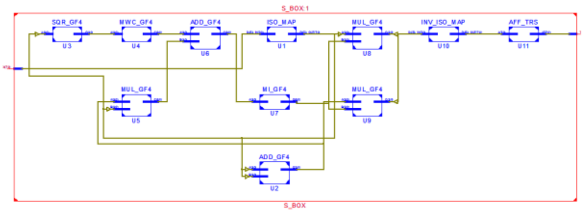

Figure 6: Main Module of S-Box

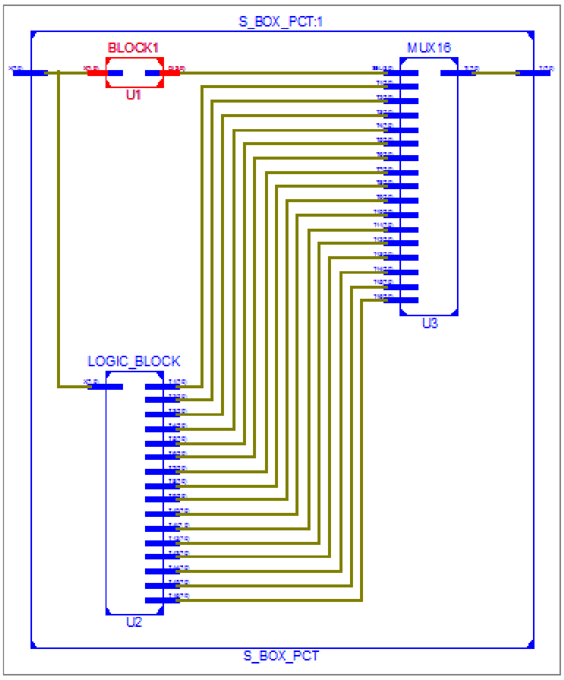

Figure 7: S-Box with PCT (Pre Computation Technique)

Xilinx ISE 12.1 is used to synthesize the design and provide post placement timing results. Table below showing the area consumed by various s-box architectures:

Table 1: Comparison of Various S-box architectures

S-BOX	SLICES	GATE COUNT
ROM-based	0	67739
PCT-BASED	168	2643
GF- BASED	58	1378

V. CONCLUSION

This paper provides an approach for FPGA Implementation of Improved S-BOX Architecture for Advanced Encryption Standard. This approach will lead to generate more secure block ciphers, solve the problem of the fixed structure S-boxes, and will increase the security level of the AES block cipher system. The proposed design is efficient in terms of area as well as speed compared to the fastest known implementation of S-box.

REFERENCES

[1] Advanced Encryption Standard (AES) – GIAC, available at : www.giac.org/cissp-papers/42.pdf

[2] Minli Dai, “Innovative Computing and Information”, International Conference, ICCIC 2011, Wuhan, China, September 17-18, 2011.

[3] Vincent Rijmen, “Efficient Implementation of the Rijndael S-Box.”, Katholieke University Leuven, Dept. ESAT, Belgium.

[4] Akashi Satoh, Sumio Morioka, Kohji Takano and Seiji Munetoh, “A Compact Rijndael Hardware Architecture with S-Box Optimization”, Springer-Verlag Berlin Heidelberg, 2001.

[5] X. Zhang and K. K. Parhi, “High Speed VLSI architectures for AES algorithm”, IEEE Transactions on VLSI Systems, Vol.12, No. 9, pp 957-967, 2004.

[6] R. Liu, K.K.Parhi “Fast composite field architectures for Advanced Encryption standard” Proceedings GLSVLSI’08, Orlando, Florida, USA, pp 65-70, May 4–6, 2008.

[7] T Good, M. Benaissa, “AES on FPGA from the Fastest to the Smallest”, LNCS 3659, pp. 427-440, 2004.

[8] J Zambreno, D. Nguyen, Alok Choudhary, “Exploring Area/Delay Trade-offs in an AES FPGA Implementation” Lecture Notes in Computer Science 3203, Proc.FPL, Antwerp, Belgium, pp 575-585, 2004.

[9] M. M. Wong, M.L.D. Wong, “A high throughput Low power compact AES S-box implementation using composite field arithmetic and Algebraic form representation”, Proc. IEEE 2nd Asia Symposium on Quality Electronic Design, pp 318-323, 2010.

[10] Rashmi Ramesh Rachh, P.V. Ananda Mohan and B.S. Anami, “Efficient Implementations of AES S box and Inverse S- box”, Proc. IEEE TENCON, Singapore, pp 1-6, 2009.

[11] M. Fayed, M. El-Kharashi and F. Watheq Gebali, “A High-Speed, Fully-Pipelined VLSI Architecture for Real-Time AES”, 4th International Conference on Information & Communications Technology, 2006.