Optimization AES S-box/Inv S-box Using FPGA Implementation

Hidayarni Hamzah[1], Nabihah Ahmad[1], M. Hairol Jabbar [2] and Chin Fhong Soon[1]
[1]*Department of Electronic Engineering, Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, Johor, Malaysia.*
[2]*Department of Computer Engineering, Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, Johor, Malaysia.*
ge160051@siswa.uthm.edu.my

***Abstract*—Advanced Encryption Standard (AES) is a common symmetric encryption algorithm and widely implemented in Wireless Local Area Network (WLAN), Radio Frequency Identification (RFID) tags and Bluetooth controller as the default choice for security services in its application. Substitution box (S-box) is a non-linear transformation and the core of AES implementation which consumed most of the power in AES hardware. This paper presents a low-complexity design methodology for the S-box/ Inverse S-box (Inv S-box) implemented in Field-Programmable Gate Array (FPGA) using composite field arithmetic and Quartus II as a tool to obtain simulation results through Verilog Hardware Description Language (HDL). This design utilized 94 slices with the hardware cost of the S-box/InvS-box is about 172 logic gates, with the power consumption of 31mW and the throughput is 1.6Gbps obtained through calculation. The design is suitable for the portable device application which requires data security with a low area and power consumption.**

***Index Terms*—AES; S-box/InvS-box; Composite Field; FPGA.**

I. INTRODUCTION

Advanced Encryption Standard (AES) is required for security services in many applications such as Bluetooth controller, Wireless Network and Radio Frequency Identification (RFID) tags. It was selected as the United State of America (USA) standard for encryption of unclassified information in 2001. As AES had announced, it replaced the Data Encryption Standard (DES) which had been the USA standards for a number of years, since 1977. AES is the currently employed specification for encrypting electronic data from the USA, National Institute of Standards and Technology (NIST) [1].

AES algorithm processes a 128-bit data blocks which are divided into 16 bytes and variable length keys of 128, 192 and 256 bits. These data bytes are mapped to a 4x4 array called the State. AES consists of four different data transformations which is Substitution bytes (SubBytes)/ Inverse SubBytes (Inv SubBytes), ShiftRow/ Inverse ShiftRow (Inv ShiftRow), MixColumn/ Inverse MixColumn (Inv MixColumn) and AddRoundkey.

S-box in a SubBytes transformation operates on individual bytes using an S-box containing a permutation of all 256 possible 8-bit values. The transformation consists of two, firstly a byte replacement, with its multiplicative inverse in the Galois field GF(2^8), using the irreducible polynomial $p(x) = x^8 + x^4 + x^3 + x + 1$ and secondly, an affine transformation over Galois Field GF(2^8). For the decryption operation, an Inv S-box is obtained by applying an Inv Affine transformation, followed by a multiplicative inversion in GF(2^8) which is controlled by multiplexer.

This proposed design focus on FPGA implementation of S-box/ Inv S-box in AES by using Verilog HDL coding. This design used composite field arithmetic. The composite field technique provides a lower area compared to Look Up Table (LUT) technique in S-box/In S-box. A few existing different construction design using composite field are found in, [2], [3] and [4]. The first efficient hardware implementation of the multiplication inversion in GF (2^8), by decomposing the finite field GF(2^8) to its sub-field GF(2^4) had proposed by [5], which leads to reduce complexity in hardware. Furthermore, Anitha N. et al. [3] proposed technique with multistage sub-pipelined architecture in order to increase the throughput of AES algorithm with the composite field by using FPGA devices. The code is VHDL that could easily be implemented on targeted FPGA devices, without changing the design.

The transformation matrix from GF(2^8) to GF($(2^2)^2)^2$ is proposed by Satoh et al. [3] and is claimed to be the most optimized hardware implementation to date, with a gate complexity of 5400 gates. Mentens et al. [11] also use the same approach as Satoh et al., [3] but with different polynomial coefficients, and achieve slightly better optimized.

The rest of this paper is organized as follows: section 2 is about the AES architecture. Section 3 describes the proposed technique composite field and its steps to construct the S-box/Inv S-box. Section 4 briefs the implementation results and the last section is the conclusion.

II. AES ARCHITECTURE

AES algorithm is an iterative algorithm which performs four transformations iteratively depending on the key length either 10, 12 and 14 times. The initial round performs AddRoundKey with the State and omits MixColumn in the final round. The decryption process performs the inverse transformation. Hardware implementation of AES included of S-box is the most expansive building block and the multiplicative inversion is the most complicated steps of the S-box transformation.

A. SubBytes/ Inv SubBytes

SubBytes is a non-linear byte substitution from the S-Box. It is done by using two methods. Firstly by using Look Up Table (LUT) and another method by using Combinational Logic. The values of the S-Box are achieved by taking

multiplicative inverse over GF(2^8) and affine transformation [12]. Ahmad N. [2] proposed an implementation of an S-box in a SubBytes transformation operates on individual bytes using an S-box containing a permutation of all 256 possible 8-bit values. The transformation consists of two, firstly a byte replacement, with its multiplicative inverse in the Galois field GF(2^8), using the irreducible polynomial $p(x) = x^8 + x^4 + x^3 + x + 1$ and secondly, an affine transformation over GF(2^8). Inv affine transformation is applied for the decryption operation to obtain an Inv S-box and followed by a multiplicative inversion in GF(2^8).

SubBytes and Inv SubBytes are using the same component. To switch the mode between SubBytes or Inv SubBytes, the multiplexer will be used as shown in Figure 1. In part III will be discussed further on how the composite field technique optimized the SubByte and Inv SuByte as illustrated in Figure 5 and 6.

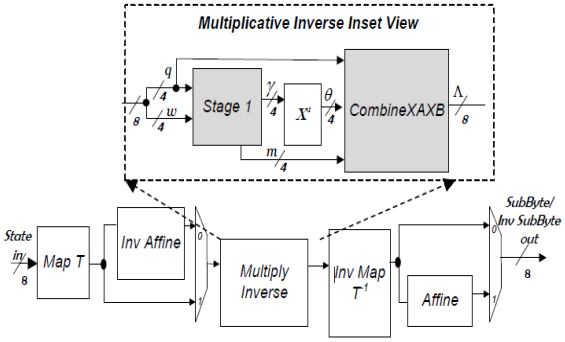

Figure 1: SubBytes and Inv SubBytes

B. *ShiftRow/ Inv ShiftRow*

In Shift Rows transformation [6], the rows of the state are shifted over different offsets. In the AES-128 algorithm, the first row remains the same and the second row is shifted to the left once. Similarly the third and the fourth row are shifted to the left cyclically by three and four shifts [8] as illustrates in Figure 2. For Inv ShiftRow the states are shifted vice versa as shown in Figure 3.

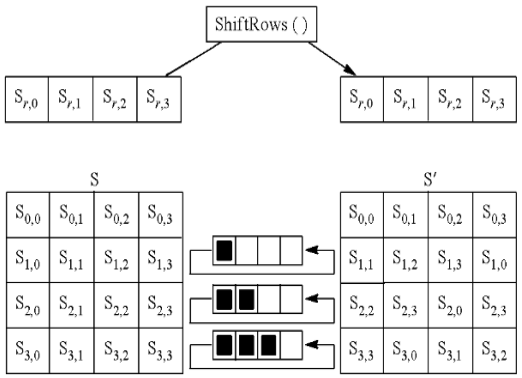

Figure 2: The ShiftRow transformation [8]

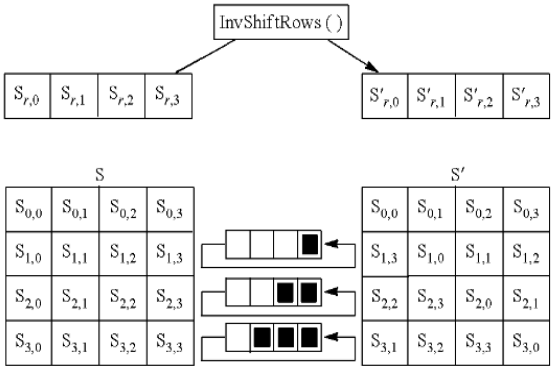

Figure 3: The Inv ShiftRow transformation [8]

C. *MixColumn/ Inv MixColumn*

In MixColumn Phase, each column of the state is multiplied by a constant polynomial over a finite field [7]. For the 128 bit key, each column is multiplied by the matrix Equation (1) which is given by,

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \quad (1)$$

In this matrix, multiplication by 1 remains unchanged, multiplication by 2 the byte will shifting to the left side and multiplication by 3 means shifting to the left side and start performing ex-or with the initial not shifted value.

The MixColumns transformation perform a polynomial multiplication over GF (2^8) modulo x^4+1 of a 4 byte x 4 byte matrix, generated by 4 right cyclic rotation (0 to 3byte locations) of the coefficients of the constant polynomial $c(x)=\{02\}_{16}x^3 + \{03\}_{16}x^2 + \{01\}_{16}x + \{01\}_{16}$ and each column vector for the state matrix. The coefficients of polynomials over GF (2^8) modulo x^4+1 are considered as the 4 bytes in each column of the phase. This leads to transform columns of the state matrix [5] given by:

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} \{02\}_{16} & \{03\}_{16} & \{01\}_{16} & \{01\}_{16} \\ \{01\}_{16} & \{02\}_{16} & \{03\}_{16} & \{01\}_{16} \\ \{01\}_{16} & \{01\}_{16} & \{02\}_{16} & \{03\}_{16} \\ \{03\}_{16} & \{01\}_{16} & \{01\}_{16} & \{02\}_{16} \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \quad (2)$$

where $S'_{0,c}$, $S'_{1,c}$, $S'_{3,c}$ are the output state after MixColumns and $S_{0,c}$, $S_{1,c}$, $S_{2,c}$ and $S_{3,c}$ are the original state

The operation of the MixColumn transformation can be understood by the following illustration as in Figure 4.

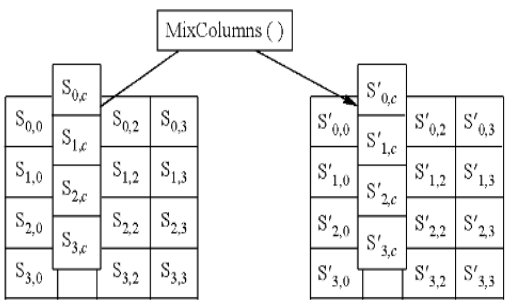

Figure 4: The MixColumn transformation

The operation of Inv MixColumn is identical to MixColumn. Each of the columns is transformed by multiplying it by a fixed multiplication polynomial [5].

D. Add Round Key

In the Add Round Key phase [8], the data is combined with the key using the ex-or operation. The key is obtained from the Rijndael's Key Schedule.

III. S-BOX/ INV S-BOX ARCHITECTURE

The S-box/Inv S-box architecture implemented combinational logic using composite field arithmetic based on [3] previous work and optimized by [2]. The architecture is implemented by using XOR circuits, Multiplexer, and basic logic gates. The composite field inversion by extending $GF(2^8)$ over GF $(((2^2)^2)^2)$ to create compact AES implementation [2].

The S-box is optimized by performing the inversion operation in a composite Galois Field of GF $(2^4)^2$ or GF $(((2^2)^2)^2)$. To optimize minimal area cost, the composite field is built by repeating degree 2 extensions under a polynomial basis by using irreducible polynomial as in Equation (2) [5] illustrates in Figure 5.

$$\begin{aligned} &GF\,(2^2) \qquad GF\,(2): x^2 + x + 1 \\ &GF\,((2^2)^2) \qquad GF\,(2^2): x^2 + x + \varphi \\ &GF\,(((2^2)^2)^2) \qquad GF\,((2^2)^2): x^2 + x + \lambda \end{aligned} \tag{2}$$

A new proposed SubByte and Inv SubByte merged the sub-component of the typical multiplicative inverse as illustrated in Figure 6 by reducing the hardware complexity of the circuit using a circuit optimization and minimization technique consists of Stage 1, the inversion and the combination of multiplication in $GF(2^4)$. Stage 1 block includes a logic optimization of multiplication in $GF(2^4)$, multiplication with constant, squaring in $GF(2^4)$, and addition included in one circuit. The block of Combine XAXB is decreased for multiplication in $GF(2^4)$ after multiplicative inversion in $GF(2^4)$. Figure 7 shown the RTL viewer generate from Quartus II software after compilation and simulation of the coding. Each block is combined by using hierarchy to become an S-box/ Inv S-box.

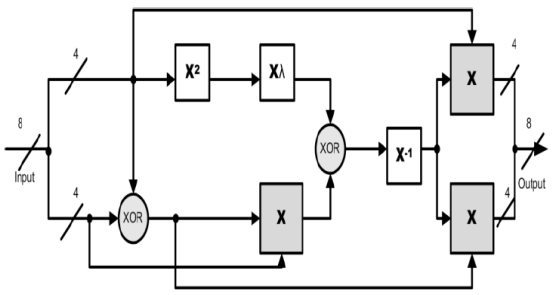

Figure 5: Block diagram of Multiplicative Inverse in $GF(2^8)$ an extension of degree 2 over $GF((2^2)^2)$ [3]

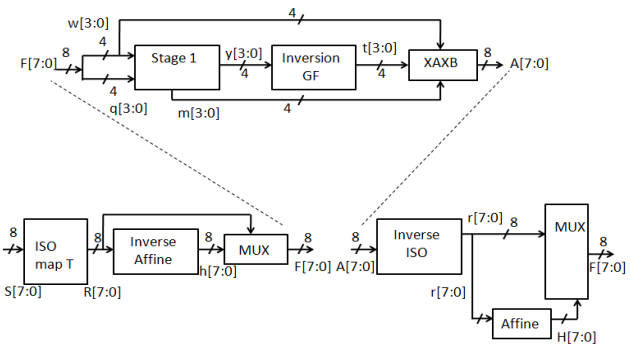

Figure 6: Multiplicative Inverse in $GF(2^8)$ transformation architecture

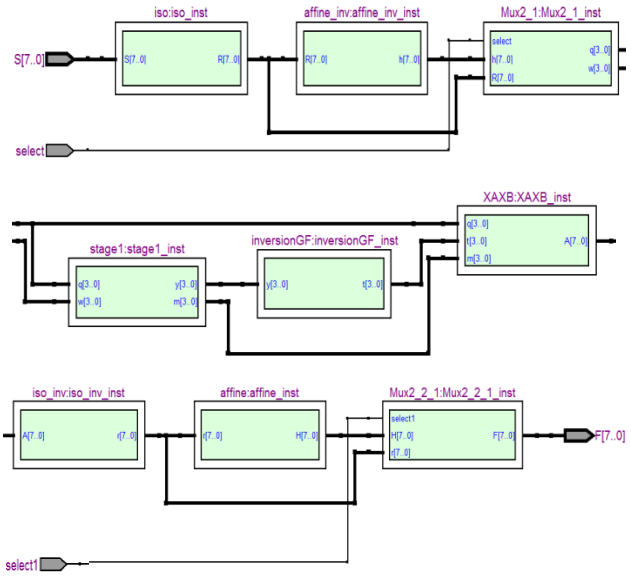

Figure 7: Overall RTL viewer generated from Quartus II

IV. RESULT AND DISCUSSION

Encryption and decryption for the proposed S-box architecture are illustrated as in Figure 6, with both S-box and Inv S-box sharing the same hardware and change the mode by switching from encryption to decryption of combination logic blocks using multiplexers.

It is an improved modification of the architecture designed by [2] using a composite field based on the polynomial basis. For this paper, the polynomial equation was transformed into HDL Verilog code perform by using FPGA device. The compile and simulation results for the design are provided from Quartus II and the throughput is calculated as Equation (3). The total equivalent gate count for the proposed S-box method is 172 gates as in Table 1 and the slices are 92 out of 4608 with the utilization of 2%. The throughput obtained through this method is 1.6Gbps.

$$\textbf{\textit{Throughput}} = \frac{\textbf{128 bit}}{(\textbf{\textit{Cycles per encrypted block x time}})} \tag{3}$$

Table 1
Total Number of Logic Gates for All Blocks

	XOR	OR	AND
ISO inv	8	-	-
Affine inv	10	-	-
Multiplexer	-	16	32
Stage1	8	-	15
Inversion GF	5	2	12
XAXB	13	-	32
ISO inv	8	-	-
Affine	11	-	-
Total Logic Gates	**63**	**18**	**91**
	63+18+91 = 172		

Table 2
Comparison of Power, Throughput, Total Logic Gates, Slices And Utilization Between Previous Work From Other Author And Proposed Method Using Verilog HDL and Quartus II with Inversion GF(24) Architecture by [2]

Author	[4]	[7]	[8]	[10]	Proposed Design
Method			FPGA		
Power (mW)	-	852	285	-	31.11
Throughput (Gbps)	22.74	-	-	-	1.6
Logic Gates	-	54	-	1650	172
Slices	1025	-	16	153	94
Utilization	-	-	1%	2%	2%

V. CONCLUSION

In conclusion, this paper presents a high throughput, low power consumption and low-area AES S-box/Inv S-box by using the architecture of composite field arithmetic from [2] and implemented with FPGA device with the Boolean expression from the architecture was coded into Verilog HDL performed by using Quartus II. The proposed design has low power consumption with low area and high throughput. It is the most optimized design compared to the result from the previous work.

ACKNOWLEDGEMENTS

This work was financially supported by FRGS Grant Vot number 1538.

REFERENCES

[1] National Inst. of Standards and Technology, "Federal Information Processing Standard Publication 197, the Advanced Encryption Standard (AES),"Nov.2001.

[2] Ahmad N. 2016. New Architecture of Low Area Aes S-box/ Inv S-box Using VLSI Implementation.2016 Penerbit UTM Press Jurnal Teknologi.

[3] Satoh A., Morioka S., Takano K. and Munetoh S. 2001. A Compact Rijndael Hardware Architecture with S-box Optimization. Advances in Cryptology — ASIACRYPT 2001. 2248: 239-254.

[4] Christy N. and Karthigaikumar P. 2012. FPGA Implementation of AES Algorithm using Composite Field Arithmetic. 2012 International Conference on Devices, Circuits and Systems (ICDCS), Coimbatore, 2012, pp. 713-717.

[5] Daemen J. and Rijmen V. 2002. The Design of Rijndael: AES - The Advanced Encryption Standard. Springer-Verlag.

[6] Ahmad N. and Hasan, R. 2012. Low-power compact composite field AES S-box/Inv S-box design in 65nm CMOS using Novel XOR Gate. Integration, the VLSI Journal, Available online ISSN 0167-9260, http://dx.doi.org/10.1016/j.vlsi. 2012. 06.002.R. C. Mikkelson (private communication).

[7] Gangadari B. and Ahamed S. 2015. FPGA implementation of compact S-box for AES algorithm using composite field arithmetic, *2015 Annual IEEE India Conference (INDICON)*, New Delhi, 2015, pp. 1-5.

[8] Prasad H., Kandpal J., Sharma D. and Verma G. 2016. Design of low power and secure implementation of SBOX for AES," *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, 2016, pp. 2092-2097.

[9] Ji J., Jung S., Jun E. and Lim J. 2009. Efficient Sequential Architecture for the AES CCM Mode in the 802.16e Standard, *2009 Second International Conference on Intelligent Networks and Intelligent Systems*, Tianjin, 2009, pp. 253-256.

[10] Ahmad N., Hasan R. and Jubadi W. 2010. Design of AES S-box using combinational logic optimization, *2010 IEEE Symposium on Industrial Electronics and Applications (ISIEA)*, Penang, 2010, pp. 696-699.

[11] Mentens N., Batina L., Preneel B. and Verbauwhede I. 2005. A Systematic Evaluation Of Compact Hardware Implementations For The Rijndael S-box. Proceedings Of The 2005 International Conference on Topics in Cryptology. 323-333.

[12] Ai W., Qing M., and Min S. 2011. Design and implementation of area-optimized AES based on FPGA, *2011 International Conference on Business Management and Electronic Information*, Guangzhou, 2011, pp. 743-746.