International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 09, September-2015

Area Optimized Architecture for AES Mix Column Operation

Neenu Shaji
Department of Electronics & Communication Engineering
Rajagiri School of Engineering and Technology
Kochi, Kerala

Bonifus P. L,
Asst.Professor
Department of Electronics & Communication Engineering
Rajagiri School of Engineering and Technology
Kochi, Kerala

***Abstract*—Advanced Encryption Standard (AES), is one of the most popular cryptographic algorithm used for data protection. The cost and power consumption of the AES can be reduced considerably by optimizing the architecture of AES. This paper proposes an implementation of the AES mix columns operation which presents a compact architecture for the AES mix columns operation and its inverse. It also proposes the method of resource sharing in case of mix column and its inverse. The delay and area consumption of the hardware implementation is compared with previous work done in this area. The proposed architecture have been implemented on the most recent Xilinx Spartan FPGA, their area and delay are compared with the previous works and it is proved that proposed technique has lower area coverage and delay.**

***Index Terms*—Security, Cryptography, AES, Encryption, Decryption, Field Programmable Gate Array (FPGA),Galois Field, RTL.**

I. INTRODUCTION

The Advanced Encryption Standard is a symmetric-key algorithm used for the encryption of electronic data. Symmetric key algorithm means the same key is used for both encrypting and decrypting the data. Due to the security it offers against attacks it has become the default choice in numerous applications.

The AES algorithm is an iterative algorithm composed of 10,12 or 14 rounds. It has a fixed block size of 128 bits, and variant key sizes of 128, 192, or 256 bits on which the number of rounds depends. The AES algorithm basically consists of four byte oriented transformation and a key expansion function.[1]

In case of 10 round process, after the initial secret key addition (roundkey (0)), the first 9 rounds are identical, with different the final round [10]. Each of the first 9 rounds consists of 4 transformations: SubBytes, ShiftRows, MixColumns and AddRoundKey. The final round excludes the MixColumns transformation. The above encryption scheme can be inverted to get a decryption structure. The SubBytes transformation is a non-linear byte substitution that operates independently on each byte of the State using a substitution table (S-box). This S-box is constructed by composing two transformations: multiplicative inverse in the finite field GF(2^8) and affine transformation[6][1].

Subbyte transformation is a nonlinear substitution that operates on individual bytes using a substitution table(sbox). Shiftrows() Is a cyclic shift of the bytes of the state with Different offsets. Add round key,a self inverting transformation transforms the input data by xoring 128-bits of the plain text with 128 bits of the expanded cipher key in the rest iteration Of the algorithm and in the subsequent iterations, the partially Processed data is xored with the expanded cipher key .In the Mix column operation, each column of the state is multiplied by the known matrix. Its a process which takes in 32 bits of Data and outputs 32 bits of data.

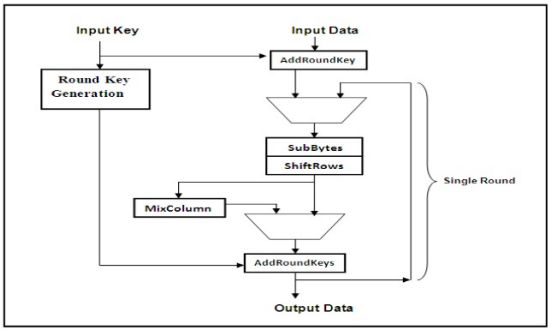

Fig. 1. Iterative Architecture of AES Encryption

By optimizing the architecture of AES we can considerably reduce the cost and power consumption of the hardware implementing AES. Among the four transformations, the sub byte and mix column operations are most computationally expensive processes.

This work addresses a method to optimize the area consumed by the mix column operation. The hardware for mix column works with 8 bit data at a time and producing 128 bit output in 16 clock cycles.This efficient architecture fits will for the embedded applications. The design is implemented in Verilog HDL and synthesized for Xilinx Spartan 3 device.The design is synthesized using Xilinx ISE tool.

II. PRELIMINARIES

The MixColumn function operates by taking four bytes as input and it outputs four bytes. Here each of the input byte affects all the four bytes of the output. A fixed matrix is used to transform the state. Each column is considered here as a four term polynomial.The columns are considered as polynomials over GF (2^8) and multiplied

modulo $(x^4)+ 1$ with a fixed polynomial $A(x) = \{03\}x^3+\{01\}x^2+\{01\}x+\{02\}$

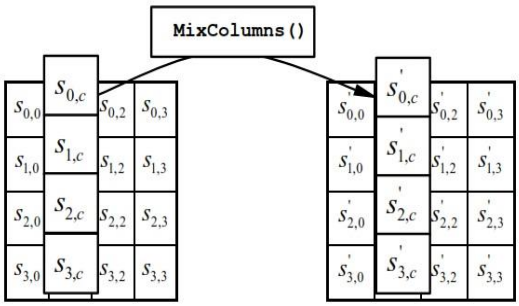

Fig. 2. Mix column Operation

InvMixColumns() is the inverse of the MixColumns() transformation.The columns are considered as polynomials over GF (2^8) and multiplied modulo $(x^4)+ 1$ with a fixed polynomial $A^{-1}(x) = \{0B\}x^3+\{0D\}x^2+\{0E\}x+\{09\}$

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Fig. 3. Mix column Fixed matrix

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Fig. 4. Inverse Mix column Fixed matrix

II. PROPOSED MIX COLUMN AND INVERSE MIX COLUMN ARCHITECTURE

The mixcolumn operation takes place in 32 bits considering one column of the state at a time. All the operations are performed in the Galois field. In galois field the addition process is performed as a XOR operation. The multiplcation by {02} in byte level is a left shift operation followed by a subsequent conditional bitwise XOR with {1B} . By repeated addition multiplication by any constant can be implemented[1].

A. Mix column

In this module,one byte of a column is treated at a time. In four clock cycles as shown in Fig.4. the result of mix column is availabale. In each clock cycle a new byte is fed to the unit, the four registers store the intermediate results of the MixColumn calculation. Every four cycles, upon the completion, the 32bit output is fed to the output registers. The architecture takes complete 16 clock cycles to complete the operation of mix column on a state[13].

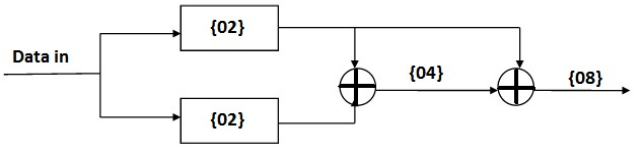

Fig. 5. Repeated addition technique

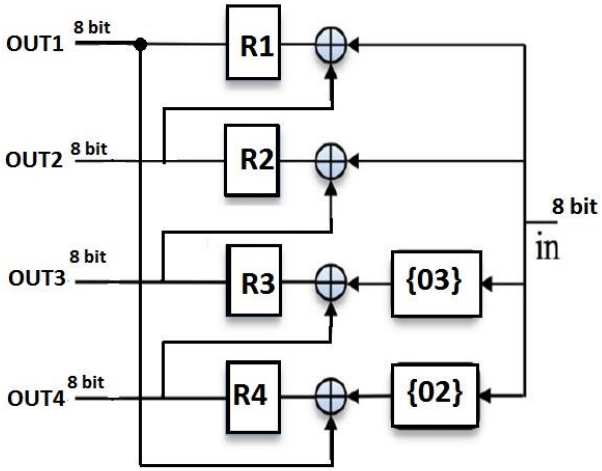

Fig. 6. Mix Column architecture

B. Inverse Mix column

InvMixColumns() is the inverse of the MixColumns() transformation. InvMixColumns() operates on the State column by column, treating each column as a four term polynomial. Here the constants in polynomial can be created similarly as in Mix column operation. The arae of it can again be considerably reduced by substrate sharing with in the units as shown in Fig.6. The unit also produces the putput in 16 clock cycles[3][4].

The area can further be reduced by sharing the units in Mix column and inverse mix column with the use of multiplexers which selects the appropriate polynomials as per the select signals[5].

C. Control unit

Since the mix column operation takes data as a column we require a control unit to provide data to the architecture as each byte and produce the output only after 4 clock cycles. The purpose of control unit is to provide enable signals to the register and mix column unit. The activity is controlled using a 3 bit counter. The values in the register R1,R2,R3,R4 in Fig.4.
is provided to the 32 bit output register at the completion of 4 cycles with the help of counter. Upon completion of the 4^{th} cycle the values in the registers are reset to zero using the enable signal provided to the mix column unit. At the 16^{th} clock cycle the mix column operation a complete state will be available. In case of 8 and 32 bit systems these operations can

take place in parallel with the other transformations of AES hence saving area and time[6].

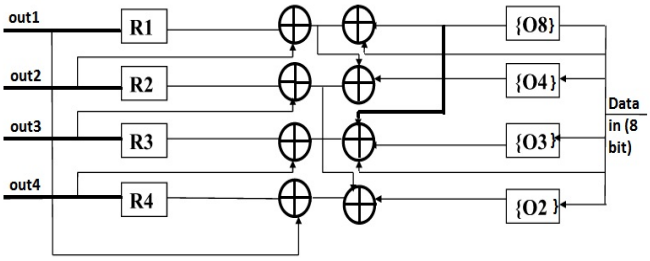

Fig. 7. Inverse Mix column Operation

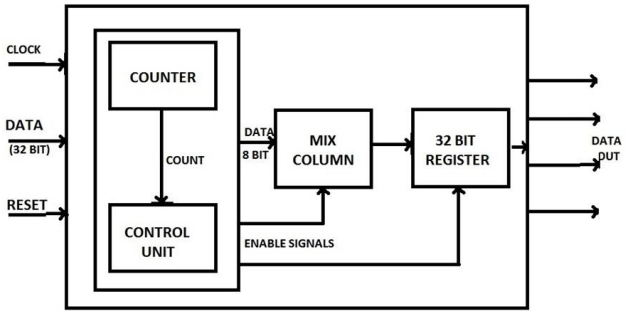

Fig. 8. Control Unit

III. IMPLEMENTATION RESULTS

The proposed Mix column and its control is implemented using Verilog Hardware Description Language in Xilinx ISE 14.6.The device utilization summary of the complete design is shown with the selected device xc3s100e-5vq100.

	No. of slices	No. of LUTs	No. of IOBs
OUR DESIGN	32	53	14
WITH EQUATION	134	2	251

Table:Device Utilization Summary

The simulation result of mix column with control unit is shown in Fig. 8.The output is verified for all combinations of the input signals.

IV. CONCLUSION

This work addresses the area optimization of the mix column architecture. The result show that the use of hardware reduces the device utilization from when mix column is implemented with equation with only delay of 16 clock cycles. The whole design is performed with the help of Xilinx and synthesized with Xilinx tools. The simulation is done in the Xilinx spartan 3 device.

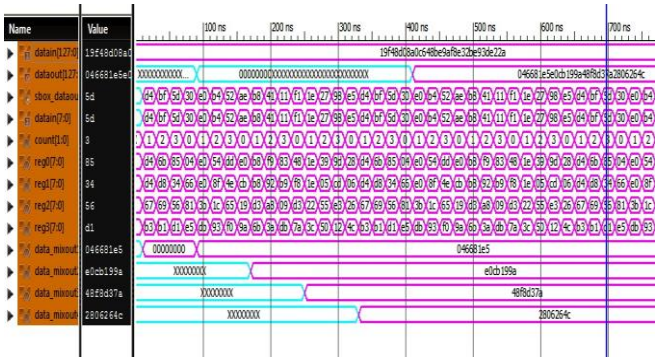

Fig. 9. Mix column Simulation result

Logic Utilization	Used	Available	Utilization
Number of Slices	32	960	3%
Number of Slice Flip Flops	14	1920	0%
Number of 4 input LUTs	53	1920	2%
Number of bonded IOBs	66	66	100%
Number of GCLKs	3	24	12%

REFERENCES

[1] P. Hmlinen, T. Alho, M. Hnnikinen, and D. Hmlinen, Design and Implementation of Low-area and Low-power AES Encryption Hardware Core.,Proceedings of the 9th EUROMICRO Conference on Digital System Design 2006

[2] National Institute of Standard and Technology., NIST FIPS PUB 197Advanced Encryption Standard, 2001.

[3] Akashi Satoh, Sumio Morioka, Kohji Takano, and Seiji Munetoh A Compact Rijndael Hardware Architecture with S-Box Optimization,SpringerVerlag Berlin Heidelberg 2001

[4] X. Zhang and K. Parhi, High-Speed VLSI Architectures for the AES Algorithm,IEEE Transactions on, vol. 12, no. 9, september 2004

[5] K. Gaj and P. Chodowiec. Very Compact FPGA Implementation of the AES Algorithm. In the proceedings of CHES 2003, Lecture Notes in Computer Science, vol 2779, pp. 319-333, Springer-Verlag.

[6] B. Liu and B M. Baas Parallel AES Encryption Engines for Many-Core Processor ArraysIEEE TRANSACTIONS ON COMPUTERS, VOL. 62, NO. 3, MARCH 2013

[7] S. Morioka and A. Satoh, An Optimized S-Box Circuit Architecture for Low Power AES Design" ,in Proc. ASIACRYPT , 2003,pp. 172186,.