ISSN 23499842(Online), Volume 1, Special Issue 1(NCEE'15), May 2015
International Journal of Innovative Trends and Emerging Technologies

AREA OPTIMIZATION FOR REDUCING CIRCUIT COMPLEXITY IN MASKED AES BASED ON FPGA

Prof.S.MURUGESWARI[1], Mrs.P.SRIDEVI[2], D.VANAJA[3]
[1]Professor & HOD, [2]AP/ECE, [3]PG SCHOLAR, [1,2,3]Dept. of ECE
[1,2,3]Sri Ramanujar Engineering College, Chennai, India.

***Abstract* – To protect the data in storage area networks from the risk of differential power analysis attacks, without demeaning a high performance, a high throughput Masked Advanced Encryption Standard (AES)engine is proposed. It uses unrolling technique which uses huge Field Programmable Gate Array (FPGA) resources. In this brief, we aim to optimize the area for a masked AES with an unrolled structure. We reduce the number of multipliers in Masked Mix Columns and Inverse mix columns. We also use FPGA block RAM to further reduce the resources.**

***Index terms* – Advanced encryption Standard (AES), differential power analysis (DPA), field programmable gate array(FPGA).**

I. INTRODUCTION

The data stored in storage area network are under the risk of information leakage in embedded applications. These applications need not only the protection at both the protocol level and the physical level but also high throughput implementations. The information leakage includes power consumption, timing and fault detection.

In 1999, Kocher et al. first overcome the drawbacks of the advanced encryption standard(AES)[1] by means of power analysis attacks. Later, the differential power analysis(DPA)[2] attack was developed as one of the most interesting power analysis attacks. From then on, numerous efforts have been devoted to the development of efficient countermeasures for the AES implementations on DPA attacks. Two representatives are the multiplicative masking and the Boolean masking. They both try to remove the correlation between the power consumption and the secret keys. The multiplicative masking can be obtained using the standard CMOS cells at the gate level or non standard CMOS CELLS . On the other hand, the Boolean masking can be easily realized at the algorithmic level and is immune to glitch attacks. The Boolean masking is more advantageous because it doesnot need extra specific hardware as in [3] and [4].

The Boolean masking is very good when applied to AES but if we directly apply it to AES, one masked AES's S-box over GF(2^8) with two 8-bit input and output masks needs to store 2^8 X 2^8 x 256 bytes. Therefore, for a whole 128 – bit Masked AES with an unrolled architecture, it needs to store around 2952.8 Mbytes. This is too big to be fit into any Field programmable gate array(FPGA). To have a reduced implementation of FPGA, one way is to transform the S-box computation of a masked AES from GF(2^8) to GF(2^4).

In this brief, we develop techniques to optimize the area of the Masked AES Mix- column unit by replacing the multipliers unit by XTIME() units. By replacing the multipliers in the AES Mix-column block we can enhance and optimize the utilization of area in the Field Programmable Gate Arrays (FPGA). The multipliers in the Mix-Column are replaced by the techniques of binary multiplication of shifting the bits towards left to multiply it by 2. The other thing to go with this is that the result is only 8-bit wide. The bit shifted out is also a thing to be considered to recover the cipher text during decryption. So we care about the bit shifted out if the shifted out bit is 1, then the result is XORed with a constant modulo operator of 1B to adjust the text to be of the required bit level.

Similarly, in the Inverse Mix Column, the multipliers are present to mix the columns it is multiplied with some constant values . These constant values are 0x02, 0x03, 0x01 in the encryption side and 0x09, 0x0B, 0x0D and 0x0E in the decryption part. These constant multiplication part stores more intermediate values in the Look Up Tables of the Field Programmable Gate Array (FPGA) which occupies more area and also consumes more power. Since the multipliers play a major role in the Mix column architecture, we have to find a correct and exact replacement for these. So to avoid the storage of intermediate values we use the XTIME() multiplication concept. As we are very much in need to optimize the area of the Field Programmable Gate Arrays, We eradicate the multiplier which adds more overhead to the Masked Advanced Encryption Standard(AES).

The rest of this brief is organized as follows. Section II presents the existing works. Section III proposes the optimized AES with detailed design methodologies about the masked Mix-Columns and Section IV shows the experimental results. Section V draws the conclusion.

II. PREVIOUS WORK

Data transformations in a storage area network usually need real-time high-throughput processing regardless of area overheads. In addition, the security issues of "data – at – rest" in a storage area network system require an add-in masking to be DPA and glitch attack resistant. Most existing works only concern high throughputs but not the ability to defend DPA and glitch attacks. Gaj and Chodowiec proposed a pipelined structure for the AES on Virtex XCV-1000 FPGA and achieved 12 Gbits/s. Standaert et al. [6] presented the design tradeoff for the further optimization of the AES implementation on FPGA platforms. Unrolling, Tiling and pipelining structures for the AES were discussed in [7]. Mc Loone and McCanny's method achieved a throughput of 12 Gbits/s

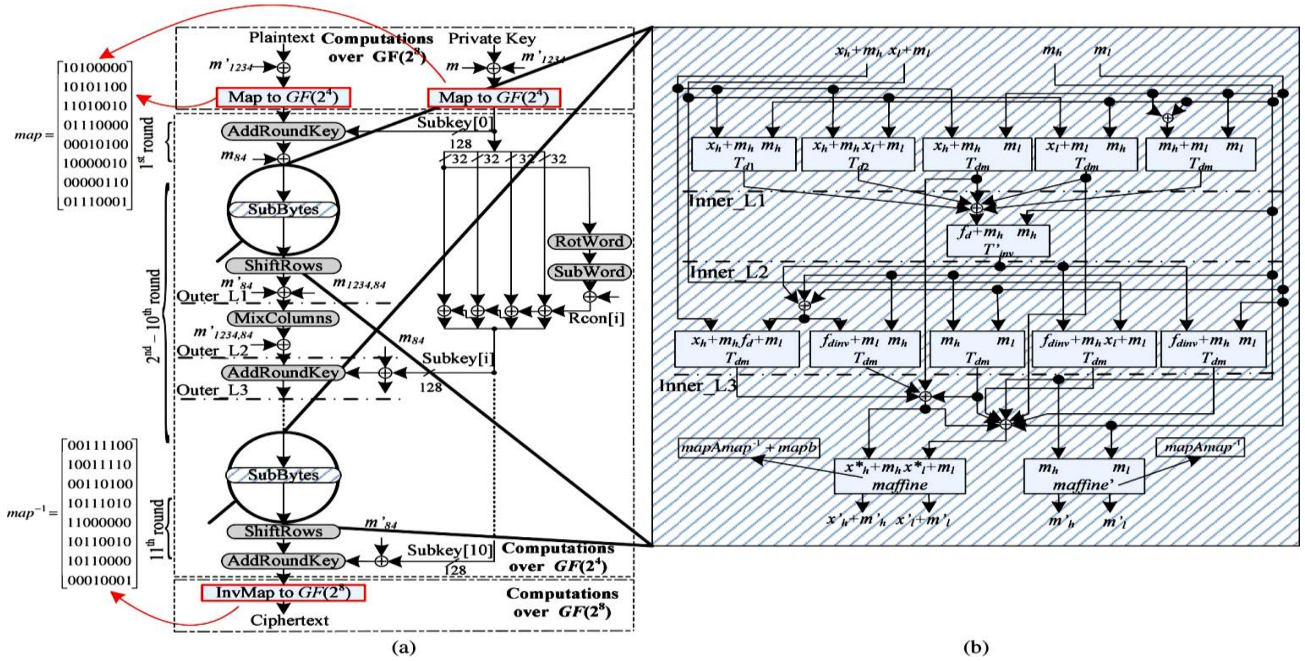

Fig. 1 Existing Masked AES with an unrolled architecture , (a) masked AES (b) Masked S- Box

using lookup table (LUT) based Sub Bytes[8]. Another approach [9] aimed at the on- the-fly generation of Sub Bytes was first proposed by Rijmen, one of the creators of the AES. Hodjat and Verbauwhede presented a fully pipelined sub Bytes architecture achieving a throughput of 21.54 Gbits/S [10]. However, all the aforementioned methods are vulnerable to DPA and glitch attacks.

Mangard et al [11] successfully broke the AES by using the DPA attack at the algorithmic level. Oswald et al. proposed a masked Sub Bytes over GF(2^4) at the algorithm level, but they only focused on software implementation [12]. Higher order masking schemes [13],[14] have been proposed. They are based on software implementations of the Mased AES.The countermeasures used in the work of Golic [15] and Canright and Batina [16] can be attacked successfully by the glitch attack [17] at the gate level.

To the best of our knowledge, no previous work has been done on the high-throughput masked AES mix columns that has the ability to optimize the area and reduce the power consumption and the delay produced in the circuits. This is due to the required huge hardware area when applying masking to AES at the algorithmic level. Most existing works are inefficient to implement masked AES with an unrolled architecture on FPGAs.

III. Proposed Masked AES for Unrolled Structure

In the Boolean masking implementation, the intermediate value x is concealed by exclusive – OR ing it with the random mask m.

In the round function of the AES, transformation.To mask a 128-bit AES, it usually needs 6-byte random values, the mask for one 32-bit Mix columns transformation. The field GF(2^8) is an extension of the field GF(2^4), over which to perform a modular reduction needs an irreducible polynomial of degree 2 $x^2 + \{1\}x + \{e\}$, and another irreducible polynomial of degree 4, $x^4 + x + 1$. In order to reduce the hardware resources, we calculate the masked AES engine mainly over GF(2^4). Fig. 1 shows the proposed masked AES, which moves the mapping and inverse mapping outside the AES's round functions. The plaintext and the masking values are mapped once from GF(2^8) to GF(2^4) and all the intermediate values are computed over GF(2^4). Finally the cipher text is mapped from GF(2^4) to GF(2^8). The adjustment of the masked Mixcolumns are discussed here.

A.Existing Masked Mix columns over GF(2^4)

Masked Mix columns can be scaled to adjust the operations over GF(2^4), and it needs to deduce the scaling factor of a modular multiplication with the fixed coefficients 0x02 and 0x03. If s is 1 byte of Mixcolumns, it holds that S = Map(Sh,Sl) ≈Shx + Sl, where S €GF(2^8) and Sh,Sl € GF(2^4). Therefore, the Scaling factors 2x+6 and 2x+7 of S equal to (4Sh+2Sl)x + (fSh+6Sl) and(5Sh + 2 Sl) + (fSh + 7 Sl). Fig 2 shows the scaling computation for the Masked Mix column.

B. Proposed Masked Mix column over GF(2^4)

In this we use the XTIME() multiplication units instead of multipliers with does the multiplication by shifting the binary bits and EXORing the result in 1B when the shifted bit is 1.In general in the encryption,the mix column is as follows.In the polynomial representation, multiplication in GF(2^8) corresponds with the multiplication of polynomials modulo an irreducible polynomial of degree 8. A polynomial is irreducible if its only divisors are one and itself, $m(x) = x^8 + x^4 + x^3 + x + 1$,or {01}{1b} in hexadecimal notation.

The modular reduction by m(x) ensures that the result will be a binary polynomial of degree less than 8, and thus can be represented by a byte. Unlike addition, there is no simple operation at the byte level that corresponds to this multiplication. The multiplication defined above is associative, and the element {01} is the multiplicative identity. For any non-zero binary polynomial b(x) of degree less than 8, the multiplicative inverse of b(x), denoted as b 1(x), and follows: the extended Euclidean algorithm isused to compute polynomials It follows that the set of 256 possible byte values, with XOR used as addition and the multiplication defined as above, has the structure of the finite field GF(2^8). The result is obtained by reducing the above result modulo m(x). If b7 = 0, the result is already in reduced form. If b7 = 1, the reduction is accomplished by subtracting (i.e., XORing) the polynomial m(x). It follows that multiplication by x (i.e.,{00000010} or {02}) can be implemented at the byte level as a left shift and a subsequent conditional bitwise XOR with {1b}.

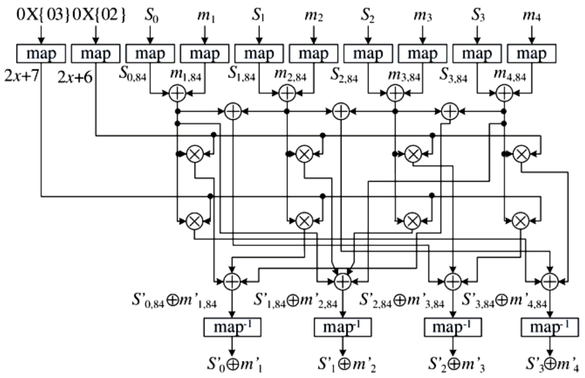

Fig. 2 Scaling Computation of the Masked Mix column

This operation on bytes is denoted by X-time(). Multiplication by higher powers of x can be implemented by repeated application of X-time ().By adding intermediate results, multiplication by any constant can be implemented.

The MixColumns transformation operates on the State column-by-column, treating each column 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), given by a(x)={03}x^3 + {01}x^2 + {01}x + {02}

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < \boldsymbol{Nb}.$$

Similarly, in the decryption end the Inv Mix Columns is the inverse of the Mix Column transformation. Inv Mix Columns operates on the State column-by-column, treating each column 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-1(x), given by a-1(x) = {0b}x^3 + {0d}x^2 + {09}x + {0e}.

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < \boldsymbol{Nb}.$$

Fig 3 shows the scaling computation for the proposed Mix column architecture where the number of multipliers are reduced and then the adders are increased to reduce and optimize the area in Field Programmable Gate Arrays. The proposed mix column architecture is reduced using X-time circuit. In the proposed method, A novel mix-column is introduced to reduce the area and power than the existing method. The proposed mix-column consists of only 12 adders and 4 X-time Circuit. Instead of normal multiplier in the mix-column, we are using X-time unit to reduce the circuit complexity.

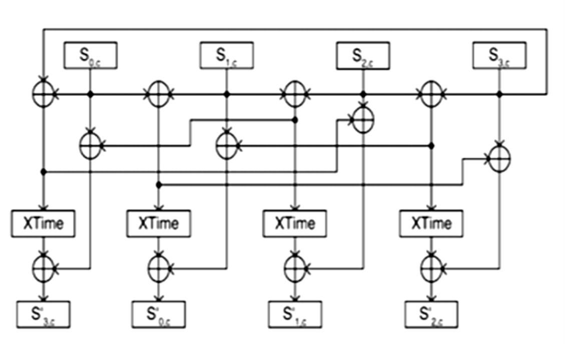

Fig 3. Proposed Mix-Column architecture

C. Optimization for Proposed Architecture

Usually, throughputs can be significantly improved by inserting pipeline registers for latency careless designs. For each masked AES's round, we insert a six-stage pipeline to enhance the throughputs. We insert three pipelines to each round of the masked AES, called outer three pipelines as shown in Fig. 1(a). The pipeline registers are inserted at the output of each transformation. We insert three pipelines to the masked S-box, called inner three pipelines as shown in Fig. 1 (b). In order to be compatible with the encryption procedure, we also insert six-stage pipelines to the key expansion in order not to affect the critical path of the main encryption.

IV. Results

In this section, we have implemented the proposed design with a very high speed integrated circuit hardware description language, synthesized our design using Xilinx ISE 13.3, and ported the design to a virtex platform.There have been some existing works on the unprotected AES for high throughput applications. The work of Mathew et al. was the fastest and smallest AES implementation on 45-nm standard cell CMOS library. The most optimized design was the work of Standaert et al., in which they achieved the best throughput among the slices on FPGA among the existing designs [6]. To our best knowledge, there have been no other works on the high – throughput masked AES on FPGA platform. Table I shows the experimental results proposed masked mix column on virtex platform.

S.NO.	PARAMETERS	EXISTING SYSTEM	PROPOSED SYSTEM
1	ENCRYPTION LUT'S	17,057	15,830
2	ENCRYPTION SLICES	9,116	8,676
3	ENCRYPTION DELAY	9.679 ns	7.859 ns
4	ENCRYPTION POWER	8.878	6.246
5	DECRYPTION LUT'S	20,471	19,438
6	DECRYPTION SLICES	10,750	10,471
7	DECRYPTION DELAY	11.456ns	7.541ns
8	DECRYPTION POWER	7.765	6.542

V.CONCLUSION

High throughput is an important factor for large data transformation systems in Storage Area networks. In order to secure "data – at – rest" and enhance the throughput, modern systems shift the encryption procedure from a software platform to a hardware platform. Hardware based encryption still opens the possibility of DPA and glitch attacks. In this brief, an LUT – based Masked Mix column has been proposed to construct the DPA- resistant design with acceptable area on FPGA.

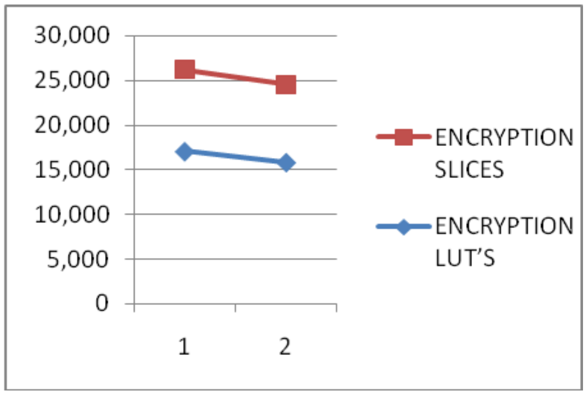

Fig. 4 Comparitive chart of the slices and LUT's in FPGA.

The proposed masked AES only needs to map the plain text and masking values from GF(2^8) to GF(2^4)once the beginning of the operation and map the cipher text back from GF(2^4) to GF(2^8) once at the end of the operation. Fig. 4 shows the comparative results of the existing and proposed system's LUT'S and Slices required area on the FPGA. Thus by removing the multipliers in the mix column and inverse mix-column architectures the overall area is reduced by more than 15%.. Also the overall power and the delay of the gates is also reduced to an extinct of 20%.Thus we achieve the proposed Mix column architecture to obtain Masked AES with the ablility to defend against DPA and glitch attacks.

REFERENCES

[1] Advanced Encryption Standard(AES), FIPS – 197, Nat. Inst. of Standards and Technol., 2001.
[2] P.Kocher, J.Jaffe, and B.Jun, " Differential Power analysis," in Proc. CRYPTO, 1999, vol.LNCS 1666, pp.388-397.
[3]L.Goubin and J.Patarin,"DES and differential power analysis(the 'duplication' method)," in Proc.CHES LNCS, 1999, vol.1717, pp. 158-172.
[4] S.Messerges,"Securing the AES finalists against power analysis attacks," in Porc.FSE LNCS, 2001, vol.1978, pp. 150-164
[5] K.Gaj and P.Chodowiec, "Fast implementation and fair comparison of the final candidates for advanced encryption standard using field programmable gate arrays", in Proc. CT-RSA LNCS, 2001, vol. 2020, pp. 84-99.
[6] F.X.Standaert, G.Rouvroy, J.J.Quisquater, and J.D.Legat, "Efficient implementation of Rijndael encryption in reconfigurable hardware: Improvements and design tradeoffs,"(in German), in Proc. CHES LNCS, 2003, vol2779, pp.334-350.
[7] G.P.Saggese, A.Mazzocca and A.G.M.Strollo, "An FPGA based performance analysis of the unrolling, tiling, pipelining of the AES algorithm," in Proc. FPL LNCS, Aveiro, Portugal, 2003, vol 2778, pp. 292-302.
[8] M.McLoone and J.V.Mc Canny, "Rijndael FPGA implementations using look-up tables," in Proc. IEEE Workshop signal Process. Syst., Antwerp, Belgium, 2001, pp.349-360
[9] A.Hodjat and I.Verbauwhede, " A 21.54 Gbits/s fully pipelined processor on FPGA," in Proc. IEEE 12th Annu. symp. Field Programm. Custom comput. Mach., 2004, pp. 308-309.
[10] S.Mangard, N.Pramstaller and E.Oswald,"Successfully attacking masked AES hardware implementations," in Proc. CHES LNCS, 2005, vol.3659, pp. 157-171.
[11] E.Oswald, S.Mangard, V.Rijmen, " A side-channel analysis resistant description of the AES S-box," in Proc. FSE LNCS, Setubal, Portugal, 2005, vol.3557, pp. 413 – 423.
[12] S.Mangard, E.Oswald, and T.Popp, Power Analysis Attacks: Revealing the secrets of Smart cards. New York: Spinger-Verlag, 2007.