International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 07, July-2015

Design of Hardware Implementable AES with Minimal Resource Utilization

K. Hemalatha,
PG scholar
Dept. of Electronics and Communications Engineering
JNTU College of Engineering, Anantapur
Anantapur, AP, India.

B. Doss,
M.Tech, (Ph.D)
Lecturer, Dept. of ECE
JNTU College of Engineering Anantapur
Anantapur, AP, India.

Abstract: **This paper presents a new method for the hardware implementation of AES algorithm. As the resource utilization plays a vital role in all hardware implementations, this paper mainly focused on utilization of minimal resources. It has been achieved with the help of Galois field transformation and masking. In this brief the operations of AES have been mapped from GF (2^8) to GF (2^2) as much as possible, so that all the operations have been performed over GF (2^2) and at last the results are transformed back to GF (2^8) from GF (2^2). In addition to this transformation Boolean masking is applied to the encription algorithm.**

I. INTRODUCTON

Physical networks such as Storage Area networks (SANs) require hardware implementable AES algorithm to ensure security. Since Field Programmable Gate Arrays (FPGAs) are attractive options for the hardware implementation of encryption algorithms, most of the early works have been worked in that way. But it requires large FPGA resources. However FPGA boards provide limited amount of resources so the design should be such that to be fit into the board.

The normal AES has been broken by Kocher *et al*. for the first time in 1999 with the help of power analysis attacks [1]. Also he stressed that the information leaked from microchips and computers has correlation with the data handled in physical devices. The power analysis attacks habitually targets one particular circuit. Hence identifying the target is the first step in power analysis. Next is estimating a hypothetical model of device's power consumption [2]. Based on that information secret key has been extracted. As most of the present FPGAs are made up of CMOS gates, the power consumption depends on number of bit transitions in the device registers. Moreover the steps involved in encryption algorithm are familiar to everyone.

With time DPA attack has become stronger. So to overcome those threats numerous works have been done for implementing AES with the ability to defend against the attacks mentioned earlier.

One of the solutions derived for the above problem is to apply masking schemes for the round function of AES algorithm. The two better masking schemes that suits the AES algorithm is

i. Multiplicative masking
ii. Boolean masking.

Realization of multiplicative masking can be done by using standard CMOS cells. But it is proven that multiplicative masking is vulnerable to DPA attacks.

Whereas the realization of Boolean masking is easier than that of multiplicative masking. So in this Boolean masking is used.

II. DEFINITION OF BOOLEAN MASKING

Boolean masking is defined as follows. In this masking scheme simple ex-or operation is performed between the random mask and the data [3]. It illustrated as below.

$$x' = x \oplus r$$

Where x is input to the mask, x' is output of the mask and r refers to random mask.

The masking operations can be explained in detailed as below. If

$$x_3 = x_1 \oplus x_2$$

Then the masked output is given as

$$x_1' = x_1 \oplus r_1$$

$$x_2' = x_2 \oplus r_2$$

$$\text{And } x_3' = x_1' \oplus x_2'$$

$$\text{Then } x_3 = x_3' \oplus r_3$$

$$\text{Where } r_3 = r_1 \oplus r_2$$

Many research works suggested that Boolean masking is resistant to DPA attacks. With the application of masking schemes, the actual data divides into multiple halves. Moreover this masked data will be distributed uniformly as compared to actual data. For this reason intruders have to estimate compound distributions for each portion of the masked data. As a result the computation complexity gets increased exponentially.

Among the round operations of AES, only the subbytes operation is non-linear. Hence the masking has to be applied outside of this function. This is illustrated in the below figure 1.

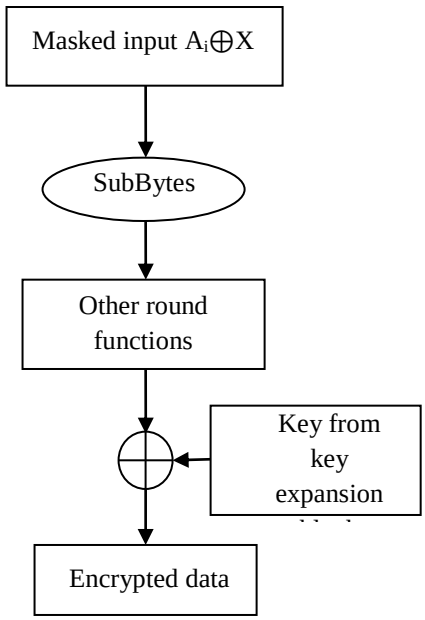

Fig 1: masking outside of subbytes step

As substitution of bytes is the only non linear operation of AES round function, care has to be taken while applying masking schemes [4].

III. CALCULATIONS ON GALOIS FIELDS

In general a field is more than just a set of elements. The field that contains finite number of elements is usually called as Galois field. Following are the operations that are performed over the GF (2^8) [5].

Let a(x), b(x) be two fourth order polynomials, the addition and multiplication operations are given as

$$a(x) = a_3x^3 + a_2x^2 + a_1x + a_0$$

$$b(x) = b_3x^3 + b_2x^2 + b_1x + b_0$$

Then the addition is

$$a(x) + b(x) = (a_3 \oplus b_3)x^3 + (a_2 \oplus b_2)x^2 + (a_1 \oplus b_1)x + (a_0 \oplus b_0)$$

That means addition operation is nothing but performing Ex-OR operation between the coefficients of like powers.

Multiplication operation is defined as

$$c(x) = a(x) \cdot b(x)$$

$$c(x) = c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0$$

Where $c_0 = a_0 \cdot b_0$

$$c_1 = a_0 \cdot b_1 \oplus a_1 \cdot b_0$$

$$c_2 = a_2 \cdot b_0 \oplus a_1 \cdot b_1 \oplus a_0 \cdot b_2$$

$$c_3 = a_3 \cdot b_0 \oplus a_2 \cdot b_1 \oplus a_1 \cdot b_2 \oplus a_0 \cdot b_3$$

$$c_4 = a_3 \cdot b_1 \oplus a_2 \cdot b_2 \oplus a_1 \cdot b_3$$

$$c_5 = a_3 \cdot b_2 \oplus a_2 \cdot b_3$$

$$c_6 = a_3 \cdot b_3$$

The result of multiplication does not have four bits like addition operation. So to convert the result modulo operation has to be performed. For AES algorithm it is gets done with the polynomial $x^4 + 1$. But on GF (2^8) the above mentioned polynomial is an irreducible polynomial. Hence the results of those operations need large memory for storage.

IV. AFFINE TRANSFORMATION

Since the computations over the field GF (2^8) require large memory resources, here comes the need of transformation from GF (2^8) to GF (2^2). This can be accomplished with the help of affine transformation.

If the operation to which masking is to be applied is a linear operation, it follows the re computation of additive constant for each new mask to facilitate the equivalence of the data transformations. But the case is different if the operation is non-linear [5].

V. PREVIOUS WORK

Yi Wang and Yajun Ha proposed a strategy and they proved that storage requirement has been reduced by 20.5%, if the operations of AES implementation are performed over GF (2^4) [6]. The block diagram is given in fig 2.

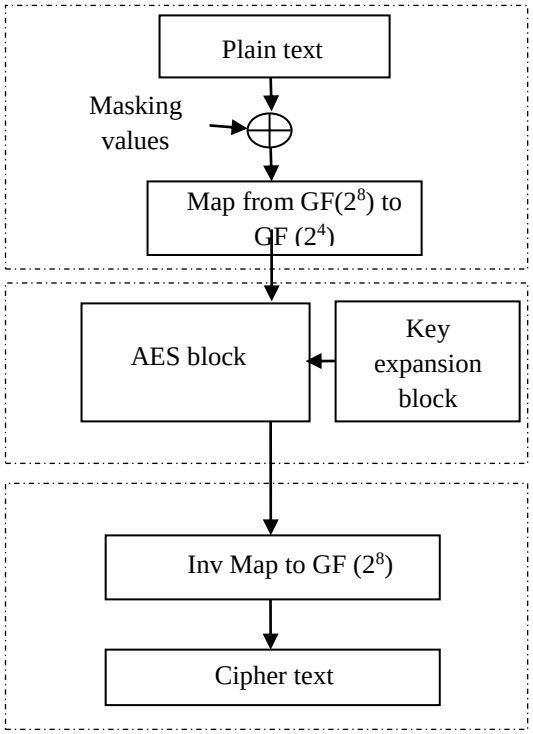

Fig 2: masked AES over GF (2^4)

Affine transformation involved in this implementation is:

Let the input values to the map function be (z+m) and m. the output values be (z+m)' and m'. Then the input belongs to GF (2^8) and the output belongs to GF (2^4).

They keep up the relation

$$(z+m+m)' = \text{map}\ (z+m+m) \qquad (1)$$

General form of affine transformation is (Ax+b), by applying this to the above equation

$$A\ (z+m+m)+b = A\ \text{map}^{-1}\ (z+m+m)'+b \qquad (2)$$

After mapping from GF (2^8) to GF (2^4)

$$\text{Map}\ (A\ (z+m+m)+b)=\text{map}(A\text{map}^{-1}(z+m+m)'+b) \quad (3)$$

The Galois field with 256 elements has irreducible polynomials which means that all the operations have to be performed over those polynomials. The two irreducible polynomials of GF (2^8) are (x^4+x+1) and (x^2+x+e). one is of degree 4 and the other is of degree 2. Poerations performed over GF (2^8) requires large memory resources to store their results. So, for that reason transformation has been done from GF (2^8) to GF (2^4). By resolving the above equations the affine transformations over GF (2^8) can be obtained. And the resultant matrices are given as

$$A=\begin{bmatrix}1&1&1&1&1&0&0&0\\0&1&1&1&1&1&0&0\\0&0&1&1&1&1&1&0\\0&0&0&1&1&1&1&1\\1&0&0&0&1&1&1&1\\1&1&0&0&0&1&1&1\\1&1&1&0&0&0&1&1\\1&1&1&1&0&0&0&1\end{bmatrix} \quad b=\begin{bmatrix}0\\1\\1\\0\\0\\0\\1\\1\end{bmatrix}$$

Affine transformation over GF (2^8)

$$A(x+m)+b+Am$$

VI. PROPOSED AES IMPLEMENTATION OVER GF (2^2)

As this paper mainly focusing on minimization of resource utilization, GF (2^2) is chosen to perform masked AES operations. GF (2^4) is an extension of GF (2^2), but because of the irreducible polynomials over GF (2^4), it has need of too big resources to store the results.

Irreducible polynomials of GF (2^4):

$$1+x+x^4\ ;\ 1+x+x^2+x^3+x^4\ ;\ 1+x^3+x^4$$

Hence to reduce the resource requirement an attempt is made in this brief. So that AES can have a feasible implementation in hardware.

The finite field GF (2^2) has four elements: 0,1, x,x+1. The primitive polynomial of the field is x^2+x+1 [7]. With these elements all the operations involved in the round functions can be performed. Addition and multiplication over GF (2^2) is somewhat similar to that of calculations over GF (2^4). Addition operation resembles Ex-OR operation. Performing Ex-OR between the coefficients of like powers gives the result of addition operation.

Usually multiplication is treated as successive addition. But the result of multiplication requires double memory resources than the actual data that is being multiplied. So modulo division is necessary to reduce the memory consumption.

In the proposed implementation Boolean masking is applied outside of the round function of AES. AES involves four key operations in order to encrypt the data. Among those operations substitution of bytes step is the only non-linear operation. The remaining three are linear. Moreover Sub-Bytes operation is treated as heart of the encryption algorithm. As substitution of byte is very familiar to everyone and plays an important role in producing cipher text, the changes made to that portion will

ensure security. In other words strength of the encryption algorithm depends on strength of Sub-Bytes step.

Besides this applying mask to a non-linear operation involves different calculations that depends on mask. As the three operations namely add-round key, shift-rows, mix-columns are linear in nature they follow the rule given here.

Shift-rows(x⊕m)=Shift-rows(x)⊕shift-rows(m)

But the non-linear operation does not follow the above rule. For non-linear operations like SubBytes step, masking follows as below.

s-box(x⊕m) ≠ s-box(x) ⊕ s-box(m)

for this reason mask is applied outside of the s-box. After applying mask affine transformation is applied to the resultant equation. Affine transformation is of the form A(x)+b. Applying this transformation on both sides of masked output gives following result.

$$q = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} r = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

The above matrix is derived from the affine transformation function over GF (2^4). Where "q=mapAmap^{-1}" and "r=mapb".

Actual equation that corresponds to the affine transformation over GF (2^4) is

mapAmap^{-1}(x+m)'+mapb+mapAmap^{-1}m'.

where (x+m) is an input to the mask, m' is a random mask and b is a constant added in the transformation function.

Figure3 illustrates the implementation of AES with optimized resource utilization. Here the masked values are mapped to the field of four elements i.e.GF (2^2). All the operations of AES are performed over there.

The expansion block plays an important role in the encryption algorithm. Normal AES involves key generation for each round of operation. Here in addition to that block key expansion is necessary as masking is applied to the actual data. The result of key expansion block contains the values produced by the EX-OR operation between actual round key and random mask. So to extract the secret key intruders need more computations which in turn changes the power estimation. At last de-masking has to be done to extract actual round key.

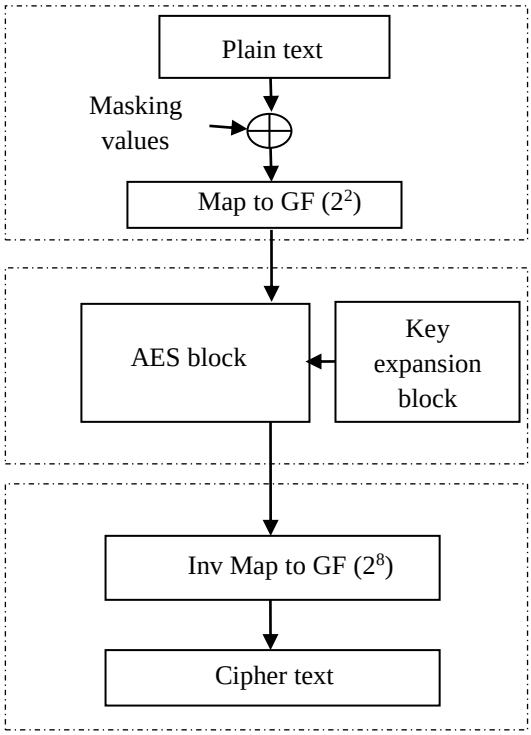

Fig 3: masked AES over GF (2^2)

In order to be compatible the order of affine transformation and masking operations has been interchanged. After applying the affine transformation by resolving those equations the matrices correspond to GF (2^2) can be obtained. They are given below.

$$q = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} r = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

Where "q = mapAmap^{-1}" and "r = mapb". While de-masking the encrypted values the same Ex-OR operation is performed. Transformation back to GF (2^8) uses matrices corresponds to the field GF (2^8).

VII. SYNTHESIS RESULTS

Table 1: synthesis report for the proposed implementation

Device Utilization Summary						
Logic Utilization	Over GF (2^4)			Over GF (2^2)		
	Available	Used	Utilization	Available	Used	Utilization
Number of slice LUTs	46560	21203	45%	46560	15406	33%
Number of slice registers	93120	376	<1%	93120	298	<1%
Number of fully used LUT-FF pairs	21460	119	<1%	21460	71	<1%
Number of bonded IOBs	240	386	160%	240	212	88%
Number of BRAM/FIFOs	156	6	3%	156	6	3%

The above tabulated results are obtained by synthesizing the logic using XILINX ISE 13.2. Hardware chosen for the synthesis is Vertex-6. By the observation of above result it is concluded that device utilization is optimized in the proposed implementation.

CONCLUSION

As area concern is important in VLSI implementations, to tackle such type of challenges an attempt is made in this paper. We have succeeded in achieving the expected result. Significant reduction in area has been showed in result.

REFERENCES

[1] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *proc. CRYPTO,* 1999, vol. LNCS 1666, pp. 388-397.
[2] F. X. Standaert, E. Peeters, G. Rouvroy, J.-J. Quisqqarter, "An Overview of Power Analysis Attacks Against FPGAs," in *proc* vol. 94 No.2, February 2006.
[3] Wing Ng, "Counter Measure for Differential Power Analysis Using Boolean Masking".
[4] Rijndael, "Advanced Encryption Standard".
[5] Jovan D. Golic and Chrystophe Tymen, "Multiplicative Masking and Power analysis of AES".
[6] Yi Wang and Yajun Ha, "FPGA Based 40.9 Gbits/s Masked AES With Area Optimization for Storage Area Network," vol. 60. No.1. January 2013.
[7] GF (2^2) from wikipedia.org/wiki/Finite_field.