International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 3, Issue 10, October 2014

Parallel AES Encryption Engine for Many core processor Arrays using Masked S-Box

Dhanya Pushkaran
M.Tech, VLSI and Embedded System
ECE, SNGCE
Kadayirippu, Kolenchery

Neethu Bhaskar
Assistant Professor
ECE, SNGCE
Kadayirippu, Kollenchery

***Abstract*— With the ever increasing growth of data communication, hardware encryption technology will become an irreplaceable safety technology. In this paper, I present a method of AES encryption and decryption algorithm with 128 bit key on an FPGA. In order to protect "data-at-rest" in memory from differential power analysis attacks with high-throughput advanced encryption standard (AES) engine with masked S-Box is proposed. By exploring different granularities of data-level and task-level parallelism, we map 2 implementations of an Advanced Encryption Standard (AES) cipher with online key expansion on a fine-grained many-core system.**

Index Terms— Advanced encryption standard (AES), differential power analysis (DPA), field programmable gate array (FPGA), masking, fine-grained, many-core, parallel processor

I. INTRODUCTION

With the development of information technology, protection of information through encryption is very important in day to day life. In 2001, national institute of standard and technology replaces the data encryption standard and select the Rijndael algorithm as the advanced encryption standard(AES)[1]. AES has been used in many applications, such as secure communication system, digital video/audio recorder, RFID tags and smart cards etc. One of the main advantage of Rijndael algorithm is that it can be used for both hardware and software implementation.

To satisfy many application numerous hardware implementation of AES has been reported to achieve high throughput even though time consuming and costly. One of the main block of AES is the SubByte transformation [1] which uses S-box look-up table that is stored in memory. This data stored in storage are under the risk of information leakage in embedded applications. The differential power analysis (DPA) attack [2] was further developed as one of the most promising power analysis attacks which is related to the power consumption. So the protection of data from DPA is very important. For that instead of using S-Box lookup table masked S-Box is being implemented. We perform the masked S-Box mainly over $GF(2^4)$. Therefore, we only need to transform the input values from $GF(2^8)$ to $GF(2^4)$ and transform the output values back from $GF(2^4)$ to $GF(2^8)$ which reduces the hardware resources.

This paper present the online expansion of two type AES implementation on a fine grained many core system to achieve high performance and throughput per unit of chip.

II. AES ALGORITHM

AES is a key iterated block cipher that contains several round of transformation on the state. It is a symmetric encryption algorithm uses 128 bit key to generate output cipher text. It takes 128 bits of data block and each 128-bit data block is considered as a 4-by-4 array of bytes, called the state. The number of iteration in the AES, Nr, is defined by the length of the round key, which are 10 for key lengths of 128 bits.

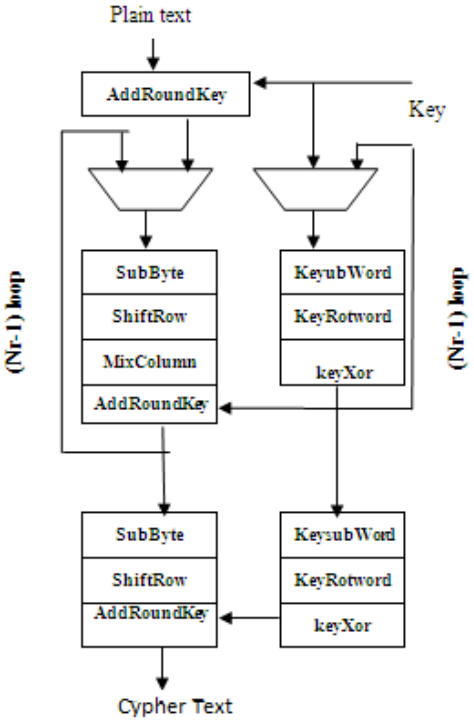

Fig 1: Block Diagram of AES Algorithm

The figure 1 shows the basic steps of AES algorithm with online key expansion. The steps include:

1. SubBytes: Nonlinear bite transformation which replace each input byte with the byte value from the substitution box. Substitution box is explained in section

ISSN: 2278 – 909X

2. ShiftRow: Each row of the state is left shifted according to the row number. First row no shifting is done, for 2nd row 1byte shifting is done and so on.
3. MixColumn: Each column of the array is considered as a polynomial over GF(2^8) and modular multiplication is done with irreducible polynomial x^4+1. The resulting polynomial is then multiplied with a fixed polynomial given in equation (1).

$$A(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} \quad (1)$$

4. AddRoundKey: Simple bitwise XOR operation of the state with the key expanded value is done. The key expansion is done by the following steps:
   1. KeySubWord: Each byte of the key value is replaced with the values from the substitution box.
   2. KeyRotWord: Each row is done a 1 byte shifting to the left.
   3. KeyXor: Each row w[i] is XORed with the previous row w[i-1] to form a new row w'[i].

III. MASKED S-BOX

In SubByte transformation, each byte is replaced with a value from S-Box. Since there are only 256 representation of 1 byte, a lookup table of S-Box can be implemented. So the power and time consumption is reduced. But this result in differential power analysis (DPA) attach[3][4].

So here S-Box using galois field can be implemented to avoid DPA attach. It can be implemented by taking the multiplicative inverse and apply the affine transformation. But calculating the multiplicative inverse in GF(2^8) is very expensive. So masked S-Box is implemented that calculates multiplicative inverse of GF(2^8) using GF(2^4). The input byte is mapped to two elements of GF(2^4) and then find out the multiplicative inverse using GF(2^4). After that the two elemnts inverse mapping to GF(2^8) is done. Figure 2 shows the steps to find out the masked s-box.

A. Multiplicative inverse

For hardware implementation far better suited representation is to see field GF(2^8) as a quadratic extension of the field GF(2^4). In this case, an element a ϵ GF(2^8) is represented as the linear polynomial with coefficient in GF(2^4)

$$\text{Map}(a) = ah\ x + al,\ a \in GF(2^8);\quad ah, al \in GF(2^4)$$

For hardware implementation, the equation for map (a) is shown in equation 2.

$$ah\ x + al = \text{map}(a), \quad ah, al \in GF(2^4),\ a \in GF(2^8) \quad (2)$$

$aA = a1 \oplus a7$, $aB = a5 \oplus a7$,
$aC = a4 \oplus a6$ $al0 = ac \oplus a0 \oplus a5$,
$al1 = a1 \oplus a2$, $al2 = aA$,
$al3 = a2 \oplus a4$ $ah0 = ac \oplus a5$,
$ah1 = aA \oplus aC$, $ah2 = aB \oplus a2 \oplus a3$,
$ah3 = aB$

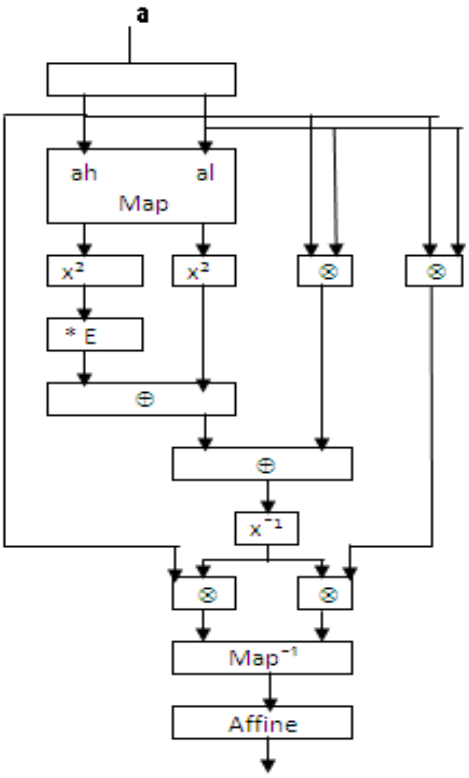

Fig 2. Block diagram of masked S-Box

After finding out the multiplicative inverse in GF(2^4), two term polynomial ah x + al converted back to element in GF(2^8). The equation for map^{-1} is shown in equation 3.

$$\text{map}^{-1}(ah\ x + al) = a, \quad ah, al \in GF(2^4),\ a \in GF(2^8) \quad (3)$$

$aA = al1\ \ ah3$, $aB = ah0\ \ ah1$
$a0 = al0 \oplus ah0$, $a1 = aB \oplus ah3$,
$a2 = aA \oplus aB$, $a3 = aB \oplus al1 \oplus ah2$,
$a4 = aA \oplus aB \oplus al3$, $a5 = aB \oplus al2$,
$a6 = aA \oplus al2 \oplus al3 \oplus ah0$, $a7 = aB \oplus al2 \oplus ah3$

Multiplication in GF(2^4) corresponds to multiplication of polynomial modulo an irreducible polynomial of degree 4. The irreducible polynomial is given by,
$M(x) = x^4 + x + 1$. For hardware implementation, byte multiplication is given in equation 4.

ISSN: 2278 – 909X

$q(x) = a(x).\ b(x).\ \mathrm{mod}\ m(x),\quad a(x),b(x),q(x) \in GF(2^4)$ (4)

$aA = a0 \oplus a3,\quad aB = a2 \oplus a3$

$q0 = a0b0 \oplus a3b1 \oplus a2b2 \oplus a1b3$

$q1 = a1b0 \oplus aAb1 \oplus aBb2 \oplus (a1\ a2)b3$

$q2 = a2b0 \oplus a1b1 \oplus aAb2 \oplus aBb3$

$q3 = a3b0 \oplus a2b1 \oplus a1b2 \oplus aAb3$

The multiplicative inverse can be find out using extended Euclidean algorithm. It can be derived by solving the equation $a(x).a^{-1}(x) \mathrm{mod}\ m_4(x) = 1$. Solution is shown in equation 5.

$q(x) = a(x)^{-1} \mathrm{mod}\ m_4(x),\ q(x), a(x) \in GF(2^4)$ (5)

$aA = a1 \oplus a2 \oplus a3 \oplus a1a2a3$
$q0 = aA \oplus a0 \oplus a0a2 \oplus a1a2 \oplus a0a1a2$
$q1 = a0a1 \oplus a0a2 \oplus a1a2 \oplus a3 \oplus a1a3 \oplus a0a1a3$
$q2 = a0a1 \oplus a2 \oplus a0a2 \oplus a3 \oplus a0a3 \oplus a0a2a3$
$q3 = aA \oplus a0a3 \oplus a1a3 \oplus a2a3$

B. Affine Transformation

Affine transformation I given by, $A' = M(a).X \oplus [v]$
Where $[v] = x^7+x^6+x^2+x$ and $m(a) = x^7+x^4+x^3+x+1$.
The equation for hardware implementation is given in equation 6.

$q = \mathrm{aff\_tran}(a)$	$q = \mathrm{aff\_trans}^{-1}(a)$ (6)
$aA = a0 \oplus a1,$	$aA = a0 \oplus a5,$
$aB = a2 \oplus a3$	$aB = a1 \oplus a4$
$aC = a4 \oplus a5,$	$aC = a2 \oplus a7,$
$aD = a6 \oplus a7$	$aD = a3 \oplus a6$
$q0 = \bar{a}0 \oplus aC \oplus aD$	$q0 = \bar{a}5 \oplus aC$
$q1 = a5 \oplus aA \oplus aD$	$q1 = a0 \oplus aD$
$q2 = a2 \oplus aA \oplus aD$	$q2 = \bar{a}7 \oplus aB$
$q3 = a7 \oplus aA \oplus aB$	$q3 = a2 \oplus aA$
$q4 = a1 \oplus aB \oplus aC$	$q4 = a1 \oplus aD$
$q5 = \bar{a}1 \oplus aB \oplus aC$	$q5 = a4 \oplus aC$
$q6 = \bar{a}6 \oplus aB \oplus aC$	$q6 = a3 \oplus aA$
$q7 = a3 \oplus aC \oplus aD$	$q7 = a6 \oplus aB$

IV. Fine Grained Many Core Architecture

The performance of architecture is roughly proportional to the square root of its complexity. So as the complexity is decreased the performance will increase but it may increase the logical area. So a many core architecture can perform better with complexity. That is instead of using single complicated core many core is used, which increases the performance.

V. AES Implementation

In this paper I present two different AES implementation with online key expansion and the throughput of the design is measured.

A. One task one processor (OTOP)

Each step in the AES algorithm is considered as a task as shown in the dataflow diagram in figure 3. Each task is mapped on to one processor in many core processors. So we call this implementation One Task One processor. For single iteration about 10 cores are required and after completing first iteration the same cores are used for the following iteration.

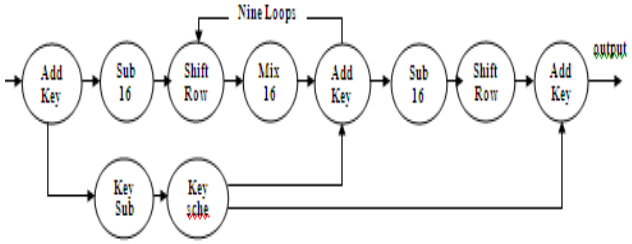

Figure3. OTOP dataflow diagram

B. Loop unrolled nine times

To enhance the throughput, new design is implemented as shown in figure 4. Here each loop is done by another set of core. So loop unrolled nine times break the data dependency and work on multiple data block. About 60 cores are required to implement this design

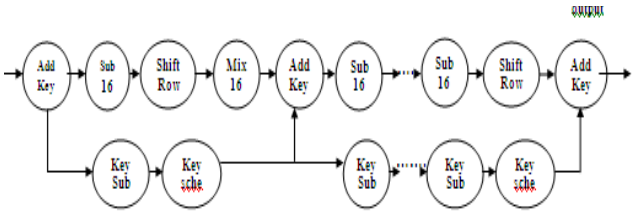

Figure 4. loop unrolled nine times data flow diagram

VI. Result

I have implemented the proposed design with hardware description language which is synthesized using Xilinx ISE 14.1and ported the design to Spartan-6 LX45 FPGA. The table 1 shows the throughput obtained from the two designs. From this table it is clear that the loop unrolled nine times design is very much faster than one task one processor design.

ISSN: 2278 – 909X

TABLE I.

Implementation	Throughput
One Task One Processor	1.98 Gbps
Loop Unrolled Nine Times	85.15Gbps

VII. CONCLUSION

Secure "data-at-rest" and enhance the throughput are the important factor for large data transformation system. so, modern systems shift the data encryption from a software platform to a hardware platform. But the hardware based encryption still facing the possibility of DPA attacks. In this case, an AES with masked S-box has been proposed to resist the DPA attach with acceptable area on FPGA. The proposed masked -Box needs to map the input values from GF(2^8) to GF(2^4) at the beginning of the operation and map the result back from GF(2^4) to GF(2^8) once at the end of the operation Which reduce about 20% area resources. By implementing the design using Loop unrolled nine times not only protect from DPA attach but also increases the throughput.

ACKNOWLEDGMENT

I would like to express my heartfelt gratitude and thanks to my beloved guide Ms. Neethu Bhaskaran, Assistant Professor, Dept. of Electronics and Communication Engineering, SNGCE Kadayiruppu, whose guidance I could complete the thesis work to the level I had planned, for the regular reviews and suggestions. It gives me great pleasure to thank her for the conviction she brought in into selecting the topic of work, and the technical and literary guidance she imparted through the different stages of its execution.

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 *Proc. FSE LNCS*, 2000, vol. 1978, pp. 150–164.
[5] S.K. Mathew, F. Sheikh, M. Kounavis, S. Gueron, A. Agarwal, S.K.Hsu, H. Kaul, M.A. Anders, and R.K. Krishnamurthy, "53 gbps Native GF(ð24Þ2) Composite-Field AES-Encrypt/Decrypt Accelerator for Content-Protection in 45 nm High-Performance Microprocessors," IEEE J. Solid-State Circuits, vol. 46, no. 4, pp. 767-776, Apr. 2011.
[6] A. Hodjat and I. Verbauwhede, "A 21.54 gbits/s Fully Pipelined AES Processor on FPGA," Proc. IEEE 12th Ann. Symp. Field-Programmable Custom Computing Machines, pp. 308-309, Apr. 2004.
[7] C.-J. Chang, C.-W. Huang, K.-H. Chang, Y.-C. Chen, and C.-C.Hsieh,"High Throughput 32-Bit AES Implementation in FPGA," Proc. IEEE Asia Pacific Conf. Circuits and Systems, pp. 1806-1809, Nov. 2008.
[8] M. McLoone and J. V. McCanny, "Rijndael FPGA implementations utilizing look-up tables," in *Proc. IEEE Workshop Signal Process. Syst.*, Antwerp, Belgium, 2001, pp. 349–360.
[9] V. Rijmen, "Efficient Implementation of the Rijndael S-Box," Dept. ESAT., Katholieke Universiteit Leuven, Leuven, Belgium, 2006. [Online] Available: http://www.networkdls.com/Articles/sbox.pdf
[10] 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.
[11] S. Mangard, N. Pramstaller, and E. Oswald,"Successfully attacking masked AES hardware implementations," in *Proc. CHES LNCS*, 2005, vol. 3659, pp. 157–171.
[12] E. Oswald, S. Mangard, N. Pramstaller, and V. Rijmen, "A side-channel analysis resistant description of the AES S-box," in *Proc. FSE LNCS*, Setubal, Potugal, 2005, vol. 3557, pp. 413–423.
[13] H. Kim, S. Hong, and J. Lim, "A fast and provably secure higher-order masking of AES S-box," in *Proc. CHES LNCS*, Nara, Japan, 2011, vol. 6917, pp. 95–107.for masked AES implementation," in *Proc. IEEE 54th Int. MWSCAS*, Seoul, Korea, 2011, pp. 1–4.
[14] M. Alam, S. Ghosh, M. J. Mohan, D. Mukhopadhyay, D. R. Chowdhury, and I. S. Gupta, "Effect of glitches against masked AES S-box implementation and countermeasure," *IET Inf. Security*, vol. 3, no. 1, pp. 34–44, Feb. 2009.
[15] E. Trichina, T. Korkishko, and K. H. Lee, "Small size, low power, side channel-immune AES coprocessor: Design and synthesis results," in *Proc. AES LNCS*, 2005, vol. 3373, pp. 113–127.
[16] S. K. Mathew, F. Sheikh, M. Kounavis, S. Gueron, A. Agarwal, S. K. Hsu, H. Kaul, M. A. Anders, and R. K. Krishnamurthy, "53 Gbps native *GF*(24)2 composite-field AES-encrypt/decrypt accelerator for content-protection in 45 nm high-performance microprocessors," *IEE*

ISSN: 2278 – 909X