Smart Intelligent Computing and Communication Technology
V.D. Ambeth Kumar et al. (Eds.)

doi:10.3233/APC210073

FPGA Implementation of Protected Compact AES S–Box Using CQCG for Embedded Applications

R.Sornalatha[a,1], N.Janakiraman[b], K.Balamurugan[a] , Arun Kumar Sivaraman[c] , Rajiv Vincent[c] and A.Muralidhar[c]
[a,1]Shanmuganathan Engineering College, Pudukkottai, India
[b]K.L.N. College of Engineering, Madurai
[c]School of CSE, VIT University, Chennai

Abstract. In this work, we obtain an area proficient composite field arithmetic Advanced Encryption Standard (AES) Substitution (S) byte and its inverse logic design. The size of this design is calculated by the number of gates used for hardware implementation. Most of the existing AES Substitution box hardware implementation uses separate Substitution byte and its inverse hardware structures. But we implement the both in the same module and a control signal is used to select the substitution byte for encryption operation and its inverse for the decryption operation. By comparing the gate utilization of the previous AES S-Box implementation, we reduced the gate utilization up to 5% that is we take only 78 EX-OR gates and 36 AND gates for implementing the both Substitution byte and its inverse. While implementing an AES algorithm in circuitry or programming, it is liable to be detected by hackers using any one of the side channel attacks. Data to be added with a random bit sequence to prevent from the above mentioned side channel attacks.

Keywords. Coupled Quadratic Congruential Generator (CQCG), Combinational circuitry S-box, Tiny AES Substitution byte and its inverse, AES, Cryptography, Random bit sequence.

1. Introduction

The coupled quadratic congruent generator is used to generate a random bit sequence, which is used to enhance the security of Substitution byte and its inverse. Modulo – 2 addition is performed between actual S-box value and the random bit sequence which is obtained using CQCG. Masked AES S-box is the resultant of the aforesaid operation. Previously S-box was implemented as a ROM table, which requires 256bytes of memory to store. This needs to be incorporated a separate memory to perform Substitution byte and its inverse operation. After performing Substitution byte and its inverse operations this memory is not useful. So, we try to reduce the

[1]Sornalatha R, Shanmuganathan Engineering College, Pudukkottai. India
Email: rslece@gmail.com

memory usage in compact embedded devices such as smart card [10]. Therefore, we choose a combinational circuitry to replace the existing S-box ROM table [1].Whenever we need a higher security for data transmission, we can select AES [12] as a cryptographic algorithm [7]. An entire work is simulated using Verilog HDL and it is implemented using Spartan – 3 FPGA device [4].

2. CFA approach in AES S – box

To optimize AES S-box combinational circuitry, we are using Composite Field Arithmetic (CFA) implementation which is mentioned in earlier works [2]. In order to achieve area reduction, we choose coupled quadratic congruential generator as a random bit sequence generator. CQCG is more secured pseudo random bit generator technique compared to other pseudo random bit generator methods like linear congruential generator, linear feedback shift register, chaotic based pseudo random bit sequence generator [5]. In order to obtain a substitution byte first we have to calculate the multiplicative inverse of the data given and next we have applied an affine transformation.

Multiplicative inverse is obtained by using Galois field (2^8) in normal basis. It is more complicated while implementing in combination circuits [6]. Instead of GF(2^8) multiplicative inversion, we use GF (2^4) multiplier, inverter as well as GF$(((2^2)^2)^2)$ multiplier, inverters. Therefore the complexity of the combinational circuit is optimized significantly [3].Both encipher and decipher in AES have same key, so it is called shared-key encryption algorithm. The data which is used in encryption/decryption of AES is in 128 bits block. Many "rounds" of processing has been done in each block of data. Four steps involved in each round of operation. 128 bits, 192 bits, or 256 bits are the allowed key sizes in AES algorithm [11]. Multiplicative inversion is the significant expensive operation in the substitution byte.

3. Computation of multiplicative inversion

The field *B* in Phase two is not built smearing a single degree-8 augmentation lead to GF (2), although spread over numerous additions of lesser steps. To decrease the cost of Phase 2 as greatly as possible, we constructed the field *B* by reiterating degree-2 extensions in a polynomial root by means of these irreducible polynomials:

$$\left.\begin{array}{l} GF(2^2) \\ GF((2^2)^2) \\ GF(((2^2)^2)^2) \end{array}\right\} \longrightarrow \begin{array}{l} : x^2+x+1 \\ : x^2+x+\varphi \\ : x^2+x+\lambda \end{array} \quad (1)$$

Where $\varphi = \{10\}_2$, $\lambda = \{1100\}_2$. The inverter over the composite field above has fewer GF (2) operators as given in equation 1 compared with the field B used in the equation 2.

$$GF(2^4) \quad : x^2 + x + 1$$
$$GF((2^4)^2) \quad : x^2 + x + \omega_{14} \quad (2)$$

Where$\omega_{14} = \{1001\}_2$.

Figure 1 depicts hardware implementation of substitution byte and its inverse.GF $((2^m)^n)$ that isbuiltby anaugmentation degree-*n* later an augmentation degree-*m*, calculating the multiplicative reciprocalsbe able to do as a group of operations above the subfields GF (2^n).Figure 2 to 4 shows the internal diagrams of proposed AES S–Box and Inverse S–Box using composite field arithmetic in normal basis.

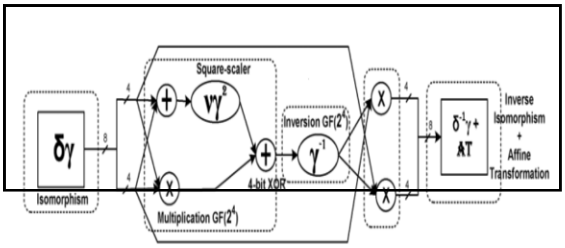


Figure 1. AES S-Box architecture.

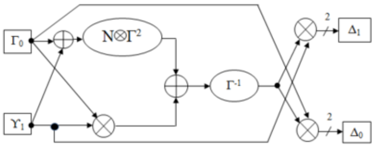


Figure 2. Normal GF (2^4) inverter.

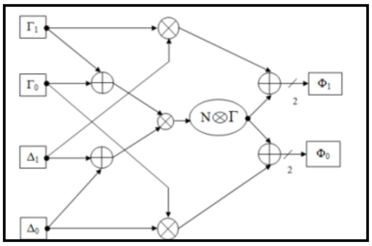


Figure 3. Normal GF (2^4) multiplier.

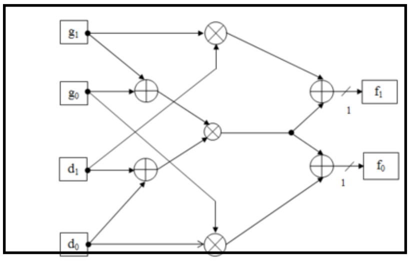


Figure 4. Normal GF (2^2) multiplier.

4. Implementation of CFA S-box

Here, we discuss about the implementation of the proposed tiny AES S-box structures [5]. Two-phase process is used for area reduction in the substitution box. The important issue while converting a ROM table into combinational circuit that should not increase the area of the circuit. To do this, the entire sub - operations in GF (2^8) inversion can be translated exclusively into logical expressions.

Our proposed tiny Substitution byte and its inverse has 78 EX-OR gates and 36 AND gates. It is 20% lesser and faster than the earlier works. Both encryption and decryption uses Substitution box and substitution box inverse respectively. They comprised of affine transformations, isomorphism functions, inverters and multiplexer. Proposed architecture is smaller in size with 78 EX-OR gates and 36 AND gates. A tiny S-Box and S-Box^{-1} was created by using composite field technique, integration the isomorphism functions with affine transformations, by means of a factoring technique, and merging the encipher and decipher routes.

Table 1. Comparisons of hardware complexities

Work Done By	Total gates		Critical Path	
	AND	**EX-OR**	**AND**	
M.M.Wong	36	96	4	2
Canright	36	91	4	2
M.Lamberger	58	110	4	1
Edwin	36	85	4	2
This Work	36	78	4	2

Evaluations of the circuit complications among this paper and the nominated earlier papers are mentioned briefly in Table 1. Table 1 shows that, the comparisons of different hardware implementations. This paper deals with a decline of 25.12% in size when compared to the work done by M.M.Wong.

5. Masked AES S-box

Arithmetical investigations which use consumed power to perform the specific task or function, radiation emitted by the source to find data approximately [9].In order to avoid the mentioned analysis hiding the data by modulo-2 addition using random variables. There are two types of masking is possible. One is addition and another one is multiplication. The best suitable mask is modulo-2 addition. Because in multiplication mask, the data zero is not hidden. So, multiplication mask is not taken into account for masking purposes [7].

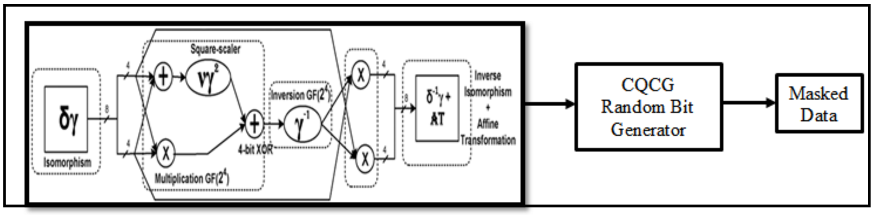


Figure 5. Architecture of the proposed system.

Figure 5 depicts the proposed architecture which is generating the substitution byte, random bit sequence using CQCG and produced a masked substitution byte as the output. Two QCG modules are coupled and their outputs are compared using particular comparator scheme. Bits are extracted from the comparator using the output of the two coupled QCG modules. A coupled QCG (CQCG) output calculated by using,

$$a_{i+1} = (P_1 a_i^2 + Q_1 a_i + R_1) \ \% \ m$$
$$b_{i+1} = (P_2 b_i^2 + Q_2 b_i + R_2) \ \% \ m \quad (3)$$

Y_{i+1} denotes the output bit of the coupled QCG,

$$\left.\begin{array}{lll} Y_{i+1} & = 1 & \text{if } a_{i+1} > b_{i+1} \\ 0 & & \text{otherwise} \end{array}\right\} \quad (4)$$

P_1, Q_1, P_2, Q_2, R_1, R_2 are assumed and a_0, b_0 are considered as base values to calculate the random bit sequence using CQCG.

Consider $P_1=13$, $Q_1=5$, $R_1=1$, $P_2=13$, $Q_2=7$, $R_2=3$, And $m=8$. $(a_0, b_0)=(7, 12)$.

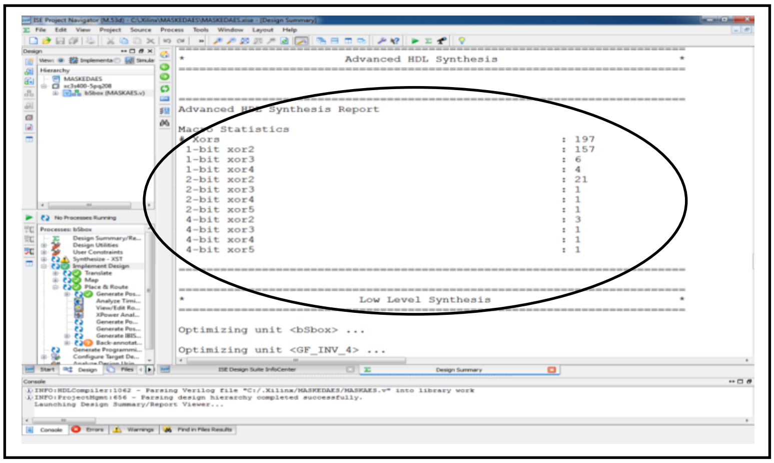
By using the equation 3, the output values of QCGs are calculated. They are random numbers. The output of two QCGs are given to comparator to produce the random bit sequence. If QCG1output is greater than the QCG2 output means the random bit output is high else the output is low.

$$\{a_i\} = (1, 3, 5, 7, 1, 3, 5, 7), \ \{b_i\} = (7, 1, 7, 1, 7, 1, 7, 1)$$

The output bit sequence Y_i is (0, 1, 0, 1, 0, 1, 0, 1)generated by the CQCG for the above input. Consider the actual substitution byte that is added with random bit sequence to produce masked data. Two independent masks are added together yields another uniformly distributed mask. Each intermediate term distribution is not dependent of the data, hence the calculation becomes secure.

6. Hardware results

Here we take Spartan 3 FPGA board to implement the compact AES Substitution byte and its inverse without masking and compact masked AES Substitution byte and its inverse[6]. We take only 78 *EX-OR* Gates utilized to get the compact AES Substitution byte and its inverse, whereas Wong's work took 96 *EX-OR* gates to implement the same.



```
=========================================================================
*                       Advanced HDL Synthesis                          *
=========================================================================


=========================================================================
Advanced HDL Synthesis Report

Macro Statistics
# Xors                                                 : 197
 1-bit xor2                                            : 157
 1-bit xor3                                            : 6
 1-bit xor4                                            : 4
 2-bit xor2                                            : 21
 2-bit xor3                                            : 1
 2-bit xor4                                            : 1
 2-bit xor5                                            : 1
 4-bit xor2                                            : 3
 4-bit xor3                                            : 1
 4-bit xor4                                            : 1
 4-bit xor5                                            : 1

=========================================================================

=========================================================================
*                         Low Level Synthesis                           *
=========================================================================

Optimizing unit <bSbox> ...

Optimizing unit <GF_INV_4> ...
```

```
INFO:HDLCompiler:1062 - Parsing Verilog file "C:/.Xilinx/MASKEDAES/MASKAES.v" into library work
INFO:ProjectMgmt:656 - Parsing design hierarchy completed successfully.
Launching Design Summary/Report Viewer...
```

Figure 6. Masked AES Substitution byte and its inverse gate utilization.

Figure 6 shows the hardware utilization of the compact masked AES Substitution byte and its inverse. We take only 197*EX-OR*Gates utilized to get the masked AES Substitution byte and its inverse, whereas can right's work took 234*EX-OR*gates to implement the same.

7. Conclusion

The Substitution byte and its inverse of Advanced Encryption Standard is implemented with least complexity of combinational circuitry using FPGA. The chip area is significant in hardware restricted distinct devices [8]. The size reduction allows the merged Substitution byte and its inverse circuit to be suitable on a chip. Hence, the pipelined and high throughput AES process on smaller chips is obtained using FPGA. Our finest tiny substitution byte and its inverse uses 25.12% lesser than the previous. For various security applications, compact version of cryptographic algorithms are required. So, in future hardware implementations of AES, this tiny merged Substitution byte and its inverse is useful. The major role of this work was the implementation of merged AES Substitution byte and its inverse that achieves a balanced structure with optimized speed and size reduction. The best design obtained has a total of *78 EX-OR* gates without masking and 197 *EX-OR* Gates to implement the Masked AES Substitution byte and its inverse. Circuit used for substitution byte and its inverse are implemented with the *18.853ns* delay.

References

[1] Wong MM, WongMLD, NandiAK, HijazinI. Construction of Optimum Composite Field Architecture for Compact High-Throughput AES S-Boxes. IEEE transactions on very large scale integration (VLSI) systems.2012 June;20(6): 1151 – 1156.

[2] Satoh A, Morioka S, Takano K, Munetoh S. A Compact Rijndael Hardware Architecture with S-Box Optimization. In: Boyd C. (eds) Advances in Cryptology — ASIACRYPT. Lecture Notes in Computer Science;2001;2248. Springer, Berlin, Heidelberg.

[3] Rudra A, Dubey PK, Jutla CS, Kumar V, Rao JR, Rohatgi P. Efficient Rijndael Encryption Implementation with Composite Field Arithmetic. In: Koç ÇK, Naccache D, Paar C (eds). Cryptographic Hardware and Embedded Systems — CHES 2001. CHES 2001. Lecture Notes in Computer Science, vol 2162. Springer, Berlin, Heidelberg.

[4] DeshpandeAM, DeshpandeMS,KayatanavarDN. FPGA implementation of AES encryption and decryption. International Conference on Control, Automation, Communication and Energy Conservation:2009: Perundurai, India: p. 1-6.

[5] Canright David,BatinaLejla. A Very Compact Perfectly Masked S-Box for AES. IACR Cryptology ePrint Archive; 2009; 11.

[6] Edwin NC Mui. Practical Implementation of Rijndael S-Box Using Combinational Logic.Texco Enterprise Ptd. Ltd Technical report; 2007.

[7] Daemen J, Rijimen V. The Design of Rijndael. Springer-Verlag; 2002.

[8] Wolkerstorfer J, Oswald E, Lamberger M. An ASIC Implementation of the AES S-Boxes. In: Preneel B. (eds) Topics in Cryptology —CT-RSA 2002. Lecture Notes in Computer Science; 2271; Springer, Berlin, Heidelberg.

[9] MassoudMasoumi. A Novel and Highly Efficient AES Implementation Robust against Differential Power Analysis. K. N. Toosi University of Tech., Tehran, Iran.

[10] WongMM, WongMLD, NandiAK,Hijazin I. AES s-box using Fermat's little theorem for the highly constrained embedded devices. 20thEuropean Signal ProcessingConference; 2012.

[11] Naga M Kosaraju, Murali Varanasi,Saraju P. Mohanty. A High-Performance VLSI Architecture for Advanced Encryption Standard (AES) Algorithm. University of North Texas Denton, TX 76203; 2008.

[12] NIST. Announcing the Advanced Encryption Standard (AES). Technical Report FIPS Publication 197, National Institute of Standards and Technology (NIST); 2001 November.