# Masked AES with Power Reduction using Pipeline Implementation for SAN

Don Samuel[1]
Department of Ece,
DhanalakshmiSrinivasan College Of Engineering
Navakkarai, Coimbatore.

M. Muthuselvam [2]
Asst.Professor
DhanalakshmiSrinivasan College Of Engineering
Navakkarai, Coimbatore

*Abstract*—A new pipelining technology based design scheme of the AES-128 is proposed. A high-throughput Masked Advanced Encryption Standard (AES) engine is used to protect data in storage area networks from the risk of differential power analysis attacks without degrading performance. The method mainly adopts the unrolling technique which requires extremely large field programmable gate array (FPGA) resources. So the principal aim is to optimize the area for a masked AES with an unrolled structure. For that, the process is mapped from $GF(2^8)$ *to* $GF(2^4)$. The masked Sub bytes, Masked mix columns, Masked add round key, Masked shift rows all are carried over $GF(2^{4)}$. To maintain the speed of encryption further the pipelining technology is applied and the mode of data transmission is modified in this design so that the chip size can be reduced. The 128-bit plain text and the 128- bit cipher key, as well as the 128-bit cipher text, are all controlled by the clock in chip. And they are divided into four 32-bit consecutive units. The 18um Complimentary MOS process shows that this new program can significantly decrease the number of chip pins and effectively optimize the area of chip. And when compared with a state-of-the-art design, this implementation reduces the area by 36.2% overall.

***Index Terms*—Advanced encryption standard (AES), differential power analysis (DPA), field programmable gate array (FPGA), masking, storage area network (SAN).**

## I INTRODUCTION

The data stored in the storage area network is always sensitive and borne to attacks by power analysis and glitches. So when applied to embedded applications it can lead to information leakage. This leakage includes timing, power consumption and fault detection.AES, a promising method was used for the protection of data's in SAN. But it was broken by Kocher *etal*in 1999 by means of power analysis attacks. Differential power analysis (DPA) was the prominent one among them. From then onwards numerous efforts were made for counter measures resulting in masking technique.

Two representatives of masking 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 is realized using either gate level standard CMOS cells or nonstandard CMOS cells. But both of them have their own disadvantages considering the attacks or the design flow demands. But on the other hand, the Boolean masking is a good candidate which is applied at the algorithmic level and also immune to DPA and glitch attacks. It is also appreciated for its easy implementation without any extra hardware's.

As reported above Boolean masking is appropriate to be applied to AES in SANS, but if applied directly can cause increase in area of the chip. For one masked AES sbox over $GF(2^8)$ with 8 bit input and output mask needs 16.8 Mbytes. Then when applied to the whole 128 bit data's it requires around 2952.8 Mbytes. This makes too big the architecture to be fit into a FPGA. To have the design feasible on FPGA the computation of S-box in masked AES is mapped from $GF(2^8)$ *to* $GF(2^4)$.

Thus in this context, to optimize the area of masked AES the whole operation is carried out in $GF(2^4)$. The operations like masked sub box, masked shift rows, masked mix columns and masked add round keys are performed in $GF(2^4)$. The input values are mapped from *GF* ($2^8$) to *GF* ($2^4$) and output values are mapped back from *GF* ($2^4$) to *GF* ($2^8$). This can reduce the overall hardware resources around 20.5%. Also inserting the pipelined registers into the round calculation and its masked S-box calculation in order to meet the requirement of high throughput for SANs.

The four operations inside each round of masked AES are again customised to reduce the area further. Each operations are pipelined. This can result in avoiding the overlap of data from successive operation and also increase the speed. Instead of waiting for the full operation in a step to complete the data's are simultaneously available to each operation beneath in each round through LUTs inside FPGA.

The rest of this brief is organized as follows. Section II provides the existing works. Section III proposes masked AES for unrolled architecture with pipeline implementation, giving the detailed design methodologies about the masked S-box and masked Mix Columns. The present technique provides excellent security of data over networks. Also explains the optimization of power consumption and area through implementing pipelining in between each round of the Masked AES. Section IV shows the experimental results. Section V presents the conclusion.

## II. PREVIOUS WORK

A SAN system requires real time high throughput implementation regardless of area overheads. Also inorder to be glitch and DPA resistant it should be masked. But the most existing works only deal with high throughput and not the attack resistance. Gaj and Chodowiec [2] proposed a

pipelined structure for the AES on Virtex XCV-1000 FPGA and achieved 12 Gbits/s. Standaert*et al.* [3] presented the design trade off for the further optimization of the AES implementation on FPGA platforms. Unrolling, tiling, and pipelining structures for the AES were discussed in [4]. McLoone and McCanny's method achieved a throughput of 12 Gbits/s using lookup table (LUT)-based SubBytes [5]. Another approach [6] aimed at 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 [7]. However, all the above mentioned methods are vulnerable to DPA and glitch attacks. Mangard et al. [8] 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 [9]. Higher order masking schemes [10], [11] have been proposed. But they are based on software implementations of the masked AES. To the best of knowledge, there is no previous work on the high-throughput masked AES that has the ability to defend against DPA and glitch attacks. This is due to the required huge hardware area when applying masking to AES at the algorithm level.

## III. PROPOSED MASKED AES FOR UNROLLED ARCHITECTURE WITH PIPELINEIMPLEMENTATON

Considering the Boolean masking process the intermediate value x is ex-ored with the mask value m.In the rounds of AES, Shift rows Mix column and Add round key are linear transformation. For a linear operation Oper, the masked Oper can be obtained by Oper($x \oplus m$) = Oper($x$) $\oplus$Oper($m$). But the nonlinear transformation S-boxis explained as S-box($x \oplus m$) ≠Sbox($x$) $\oplus$S-box($m$). so for masking the nonlinear operation a new s-box is created S-box*'*, where S-box*'*($x$ *xorm*) =Sbox($x$) *xorm'*, where $m$ and *m'* are the input and output masks of Sub Bytes. To mask a128 bit AES, there should be 6 random values.

The 128 bit plain text data is first xor'd with the 128 bit round key using carry lessGalois field additions. After this performance and power depended s-box block computes the multiplicative inverse of each byte in parallel. Now the bytes undergo linear interpolation in the mix column block, where each byte is multiplied by scaling factors and adjacent bytes are added to get 128 bit round output. These are all done in *GF(2^4)*. Fig.1 shows the masked AES, which moves the mapping and inverse mapping outside the AES's round function. The process for the masked Sub Bytes and masked Mix Columns is discussed in the following, and the masked Shift Rows and masked AddRoundKey remain the same.

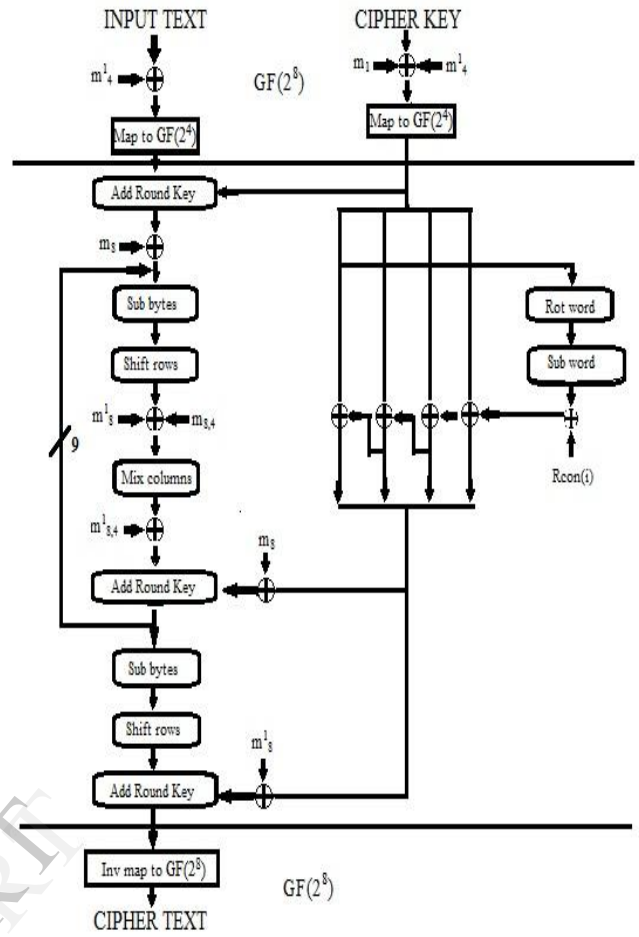


Fig.1 Masked AES with an unrolled architecture

### A. *Masked S-Box Over GF*($2^4$)

At first S-box computes $X^{-1}$ from X, such that $X.X^{-1}=1$. Followed by that is affine transformation (A. $X^{-1}$ + b) involving a matrix multiplication and addition of constant 'b'. This computation in GF($2^8$) is expensive. So S-box input in GF($2^8$) is represented as sh . x + sl, where sh and sl are 4bit elements of GF($2^4$). This conversion is made by applying a mapping transform (map). The inverse calculated in GF($2^4$) using GF($2^4$) multiplications, additions squaring and inverse. Finally it is mapped back into GF($2^8$) by applying inverse map ($map^{-1}$). Affine transform is used now to generate final s-box output. In Fig.2, the s-box map operation is the mapping transformation of 8 × 8 matrix, and $map^{-1}$ is constructed by the inverse map operation. The input values of the map function are ($z$ + $m$) and $m$, and the output values of the map function are ($z$ + $m$)*'*,and$m'$, where *[*($z$ + $m$),$m$*]*€ *GF*($2^8$) and *[*($z$ +$m$)*'*,$m'$*]* € *GF*($2^4$). Specifically it holds that

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

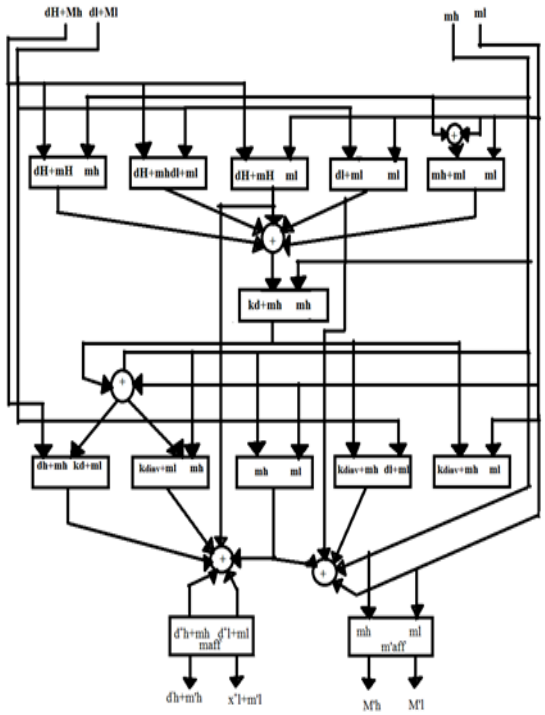

Fig 2.Masked S-box

The maffandmaff*'* are used for scaling the output values and the output values. To obtain the scaling values, the normal affine function (*ad*+ *b*) can be applied to the left and right side of above equation (1) resulting in obtaining

$$\text{maff} = \text{map}A\text{map}^{-1} + \text{map}b \quad (2)$$

$$\text{maff}' = \text{map}A\text{map}^{-1} \quad (3)$$

The mapped s-box inversion is carried out in GF($2^4$). For that there are four pre computed tables $T_1$, $T_2$, $T_3$ and $T_4$. These tables take input as 8 bits. They perform addition, squaring, multiplication and inversion respectively. There are two other tables Tmp and $Tmp^{-1}$ which do isomorphic mapping from GF($2^8$) to GF($2^4$) and return back GF($2^4$) to GF($2^8$). The overall design of the masked AES S-box depicted in Figure b requires two inputs of 8 bits each, the data which is masked and the mask that is used at present, and it produces two outputs, which is also of 8 bits each: the masked version of the sbox transformation and the improved mask.

*B. Masked Mix Columns Over GF*($2^4$)

This sub block of Masked AES involves linear interpolation of individual bytes where they are multiplied using scale factors and added together. Here the scale factor, S used for multiplication with fixed coefficients 0x02 and 0x03 are found as:. IfSis a byte of Mix Columns, it leads to*S*= map(*Sh*, *Sl*) = *Shx*+ *Sl*, where *S* €*GF*($2^8$) and *Sh*, *Sl* € *GF*($2^4$). Therefore, the new scaling factors 2*d* + 6 and 2*d*+ 7 of *S*is equal to (4*Sh* + 2*Sl*) *d*+ (*fSh*+ 6*Sl*) and (5*Sh* + 2*Sl*) *d*+ (*fSh*+ 7*Sl*).Fig.3 explains the computation of scaling factors for masked mix columns.

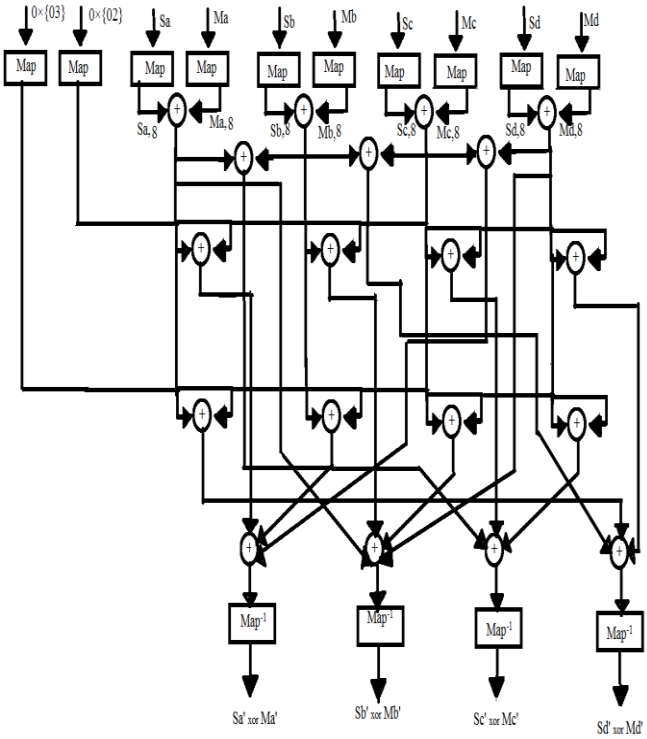

Fig 3.Scaling computation of masked mix columns

*C. Optimizing area and power with 3 stage Pipeline Implementation*

Here a block based top-level implementation of the design for new encryption process is proposed. Round 1 to Round 10 represents the individual round in the AES encryption. The inclusion of pipelining between each of the rounds will achieve a high performance, high productive encryption implementation. Although implementing a repetitive pipelining based approach is an option, for ensuring the clarity and simplicity, a fully expanded implementation for all rounds including pipelines are done. The data generated in each individual round is successively utilized as the input in the next round. Block level view of a single round implementation with internal pipelining is exploiting the loop level parallelism which the AES offers, determined that asimple ten stages pipelining of the top level withpipelined lower level blocks of the hierarchy is all that is needed to achieve high throughput. This is one of the easiest methods where high performance can be achieved in a very minimal amount of time, thus reducing the overall design implementation cycle.There is a pipeline stage between each round as explained in fig.4. However, our internal (inside each round) pipeline design is already implemented in the Masked AES process. Therefore In each round new pipelines are introduced in between the four operation Masked Sub-byte, Masked Shift rows, Masked Mix columns and at last Masked Add round key. The design has three pipeline stages. The First one after byte-sub

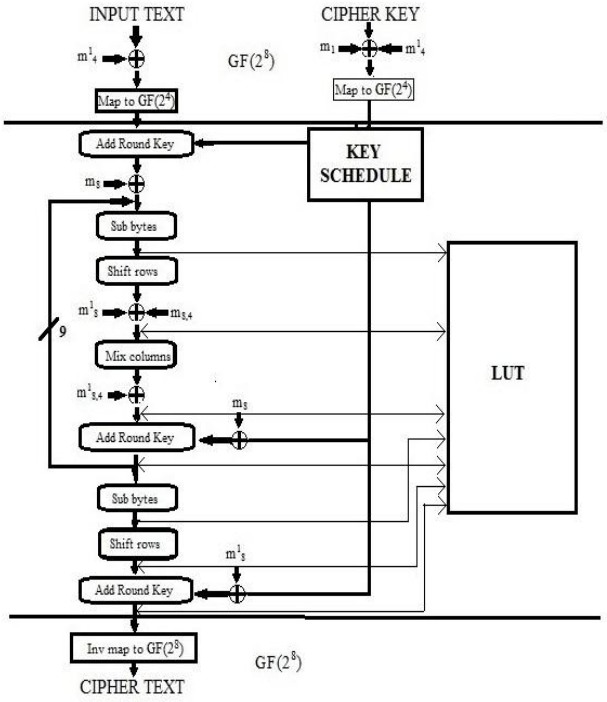


Fig. 4.Masked AES with pipeline implementation

operation, second after shift-row operation, and the last before data output. In each round, the design has four or seven pipeline stages, the first one after sub-byte operation and three or six in a sub byte operation. In addition the proposed design has one pipeline stage in key generation.Internally, the key expansion block is named itself as a pipelined implementation between the key operations from round 1 to round 10. This is automatically realized by the parallel nature of the design.

## IV. RESULTS

In this section, method implements the proposed design with a very high speed integrated circuit hardware description language, synthesized our design using Xilinx ISE 14.1 and it is ported to a Virex-7 low voltage XC7VX330TL platform. The area-optimized design with the pipelined structure proposed in this brief achieves36.2% area reduction.The objective is optimization of the design for area and power together with speed and performance. Looking for the power consumption, Masked AES without pipelining has a consumption of 143 mw of power.But the proposed method with 3 stage pipelining has only 117 mw of power consumption as in fig 5 and 6 of table 1. Also the former method bears a delay of 2.598 ns but the latter one has only 1.859 ns which again increase speed of the system.

Considering the table 2 and 3, methods are compared for area, throughput and delay. The method proposed with 3 stage pipelining get advantageous than the previous method and of course all the others too. Comparing masked AES with and without pipelining gives clean merits for the one with pipelining. Area overhead is reduced considering the no of LUTs used by each of them. Even though the LUT FF pair is increased no of LUTs plays the major role. Thus the pipelining does no harm to the no of LUTs used.Thus the technique employed makes the cipher text resistant.

TABLE.1.
Result comparison for on chip power

| D | E | F | G | H |
|---|---|---|---|---|
| On-Chip | Power (W) | Used | Available | Utilization (%) |
| Clocks | 0.000 | 1 | --- | --- |
| Logic | 0.000 | 3802 | 204000 | 2 |
| Signals | 0.000 | 3543 | --- | --- |
| IOs | 0.000 | 262 | 600 | 44 |
| Leakage | 0.143 | | | |
| Total | 0.143 | | | |

Fig.5 For masked AES without pipelining

| D | E | F | G | H |
|---|---|---|---|---|
| On-Chip | Power (W) | Used | Available | Utilization (%) |
| Clocks | 0.000 | 1 | --- | --- |
| Logic | 0.000 | 1374 | 204000 | 1 |
| Signals | 0.000 | 2327 | --- | --- |
| BRAMs | 0.000 | * | * | * |
| IOs | 0.000 | 391 | 600 | 65 |
| Leakage | 0.117 | | | |
| Total | 0.117 | | | |

Fig.6 For masked AES with pipelining

TABLE.2.
Results comparison of the Masked AES without pipelining and with pipelining

| **DEVICE UTILIZATION SUMMARY (ESTIMATED VALUES)** | | |
|---|---|---|
| **TECHNIQUES** | ***Masked AES without pipelining*** | |
| ***LOGIC UTILIZATION*** | ***USED*** | ***AVAILABLE*** |
| No of Slice LUTs | 3794 | 204000 |
| No of fully used LUT-FF pairs | 512 | 4007 |
| No of Slice Registers | 786 | 408000 |
| No of Block RAM/FIFO | - | - |
| No of BUFG/BUFGCONTROL | 1 | 32 |
| No of bounded IOBs | 262 | 600 |

| **DEVICE UTILIZATION SUMMARY (ESTIMATED VALUES)** | | |
|---|---|---|
| **TECHNIQUES** | ***Masked AES with 3 stage pipelining*** | |
| ***LOGIC UTILIZATION*** | ***USED*** | ***AVAILABLE*** |
| No of Slice LUTs | 1388 | 204000 |
| No of fully used LUT-FF pairs | 634 | 1525 |
| No of Slice Registers | 952 | 600 |
| No of Block RAM/FIFO | 5 | 408000 |
| No of BUFG/BUFGCONTROL | 1 | 750 |
| No of bounded IOBs | 391 | 32 |

TABLE.3
Results comparison of the Masked AES using pipeline with others

| ***Authors*** | ***Methods*** | ***Latency cycles*** | ***Area (slices/lut)*** |
|---|---|---|---|
| MANGARD | Protected 6 stages pipelined | 66 | 14219/ 42594 |
| BASELINE | Unprotected 6 stages pipelined | 66 | 5927/ 17771 |
| YI WANG YAJUN HA | Protected 6 stages pipelined | 66 | 10155/ 33517 |
| PROPOSEDW ORK | Protected 3 stages pipelined | 66 | 1388/ 204000 |
| ***Authors*** | ***Platform*** | ***Throug hput*** | ***Delay*** |
| MANGARD | XC6VLX240T | 26.513 | - |
| BASELINE | XC6VLX240T | 44.047 | 2.68 ns |
| YI WANG YAJUN HA | XC6VLX240T | 38.68 | 2.85 ns |
| PROPOSEDW ORK | XC7VX330TL | 40.8 | 1.859 ns |

## V. CONCLUSION

High throughput is an important factor for large data transformation systems in SANs. In order tosecure "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. The masked AES only needs to map the plaintext and masking values from $GF(2^8)$to $GF(2^4)$ once at 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.

Therefore, by moving the mapping and inverse mapping outside the masked AES's round function, it can reduce 20.5% area resources. A simple ten stage pipelining of the top level with pipelined lower level blocks of the hierarchy is all that is needed to achieve high throughput. It achieves less power consumption for the proposed masked AES with the ability to defend against DPA and glitch attacks using the pipelined implementation.

## ACKNOWLEDGMENT

The author would like to thank the college management, and Faculty Members, of Department of Electronics and Communication Engineering, DhanalakshmiSrinivasan College of Engineering, Navakkarai, Coimbatore for many insightful discussions and the facilities extended for completing the task

## REFERENCES

[1] Yi Wang and Yajun Ha, "FPGA-Based 40.9-Gbits/s Masked AES With Area Optimization for Storage Area Network," in ieeetransactions on circuits and systems—ii: express briefs, vol. 60, no. 1, january 2013

[2] 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.

[3] F. X. Standaert, G. Rouvroy, J. J. Quisquater, and J. D. Legat, "Efficient implementation of Rijndael encryption in reconfigurable hardware: Im-provements and design tradeoffs," (in German), in *Proc. CHES LNCS*, 2003, vol. 2779, pp. 334–350.

[4] M. McLoone and J. V. McCanny, "Rijndael FPGA implementations uti-lizing look-up tables," in *Proc. IEEE Workshop Signal Process. Syst.*, Antwerp, Belgium, 2001, pp. 349–360.

[5] V. Rijmen, "Efficient Implementation of the Rijndael S-Box," Dept. ESAT., KatholiekeUniversiteit Leuven, Leuven, Belgium, 2006. [Online]. Available: http://www.networkdls.com/Articles/sbox.pdf

[6] A. Hodjat and I. Verbauwhede, "A 21.54 Gbits/s fully pipelined processor on FPGA," in *Proc. IEEE 12th Annu. Symp. Field-Programm. CustomComput. Mach.*, 2004, pp. 308–309.

[7] S. Mangard, N. Pramstaller, and E. Oswald, "Successfully attacking masked AES hardware implementations," in *Proc. CHES LNCS*, 2005, vol. 3659, pp. 157–171.

[8] 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.

[9] 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.

[10] C. Carlet, L. Goubin, E. Prouff, M. Quisquater, and M. Rivain, "Higher-order masking schemes for S-boxes," in *Proc. FSE LNCS*, 2012, vol. 7549, pp. 366–384.

[11] Z. Yuan, Y. Wang, J. Li, R. Li, and W. Zhao, "FPGA based optimization for masked AES implementation," in *Proc. IEEE 54th Int. MWSCAS*, Seoul, Korea, 2011, pp. 1–4.

[12]*Advanced Encryption Standard (AES)*, FIPS-197, Nat. Inst. of *standards*and Technol., 2001