A High Performance, Low Energy, Compact Masked 128-Bit AES in 22nm CMOS Technology

Yuan-Hsi Chou*
University of Michigan
Ann Arbor, MI USA
yuanhsi@umich.edu

Shih-Lien L. Lu
TSMC
Hsinchu, Taiwan ROC
slluc@tsmc.com

***Abstract*—Advanced Encryption Standard (AES) is a specification for electronic data encryption. This standard has become one of the most widely used encryption method and has been implemented in both software and hardware. AES has excellent resistance against linear and differential cryptanalysis. Although the standard itself is algorithmically secure, based on the implementation, it can be vulnerable to attackers through side channels. For example, it has been shown that by measuring the implementation's power and performing statistical analysis on multiple traces the secret key used can be unveiled. This paper presents an efficient hardware based 128-bit AES design using a masking scheme which is resistant to a side channel attack. This masked design is implemented in TSMC 22nm technology. The resulting implementation is high in performance, low in energy and silicon area. It can run at more than 400MHz translating to a throughput of 5.12Gbps. The total area of the AES block is 0.0169mm^2. The energy consumption is at 9.77pJ/bit or 1.25nJ/block.**

Keywords: AES encryption, data masking, side channel analysis, DPA, CPA, countermeasure, hardware implementation, efficient CMOS design

I. INTRODUCTION

It has become a pressing issue to ensure Internet of Things (IoTs) are secure and trusted as they are widely used and gain popularity. Compromised IoTs may lead to loss of confidentiality, integrity of not only the device but also the confidentiality of server systems behind used to support them. Making IoTs resistant to hardware attacks is challenging since these devices have easier physical access than servers housed in data-centers. Cryptographic-system is an important part of total security. It is used to protect not only the data communicated but also the system itself. Commonly used cryptosystems are secure algorithmically. In general, hardware implementation of encryption for standard security protocols, when implemented correctly, is not only more efficient in energy but also harder to attack than their software counterpart. However, if we are not careful about the implementation of the cipher, it can leak information through side channels attacks compromising the theoretical strength of the security protocol.

The Advanced Encryption Standard (AES) is a FIPS-approved cryptographic algorithm [1] that can be used to protect electronic data. Although AES has excellent resistance against algorithmic attacks it is vulnerable under side channel attacks (SCA) [2]. A well cited SCA named Differential Power Analysis (DPA) [3] has brought much attention in the literature. DPA is a more advanced form of side channel attack that can allow an attacker to discover the intermediate values within cryptographic computations through statistical analysis of data collected from multiple cryptographic operations. DPA has been shown in practice on ASIC AES implementation in details [4].

Masking is a widely used DPA countermeasure as the intermediate values in the cryptographic algorithm are no longer correlated to the original internal values. A previously proposed masking method is provably secure [5]. By performing a function on the original input with a randomly generated mask, we are able to protect the design against DPA based on Hamming weights. This is because correlation has been scrambled. Simple first order DPA will not be able to infer information on the secret key.

In this paper, a high performance, low energy, compact, masked 128-bit AES is implemented in TSMC 22nm technology. We demonstrate through simulation that its resistance against DPA attack while incurring no performance loss. This design is very efficient in energy and area and is suitable for IoTs. The rest of the paper is organized as follows. In section II we briefly describe the needed background on AES. In section III we introduce a compact masked AES implementation based on [6]. We also cover how fresh masks are generated each round and the detail design of the masked S-Box. In section IV we presented the implementation results and showed that this design is free from DPA. In section V we briefly discuss how this design can be improved. Finally, we draw the conclusion.

II. AES CORE DESIGN

A. Background - AES Encryption Algorithm

AES is a symmetric cryptosystem based on a substitution permutation network. It is a block cipher which has a fixed plaintext of 128 bits and key size of either 128, 192, or 256 bits. It has 10, 12 or 14 rounds depending on the key length. The encryption process starts with *KeyExpansion* which takes the input key and expands it into 10, 12, or 14 additional *RoundKeys* depending on the key bit length using the Rijndael's key schedule. An initial encryption round is performed using the original key and is followed by 10, 12, or 14 encryption rounds. Each round consists of 4 steps (layers): *SubBytes, ShiftRows, MixColumns,*

* work performed while interning at TSMC

978-1-7281-0655-7/19/$31.00 ©2019 IEEE

and *AddRoundKey*. AES processes data in byte sized chunks represented as a 4x4 matrix and the complete set of those data bytes is called the state. *SubBytes* involves replacing each byte of the state with a 8-bit substitution box called the S-box which is a non-linear transformation. *ShiftRows* cyclically shifts the each row of the state matrix by a certain offset. In *MixColumns*, the four bytes of each column is combined with an invertible linear transformation. Lastly, *AddRoundKey* simply adds or XORs the current state with the *RoundKey*. In the final round, the *MixColumns* step is omitted and the cipher text is obtained after completing the *AddRoundKey* step in the final round. The complete process is shown in Figure 1 below (where Nr is 10, 12 or 14). Although decryption is not shown, its process is similar to encryption. Basically the steps are performed in the reverse order with the corresponding inverse functions.

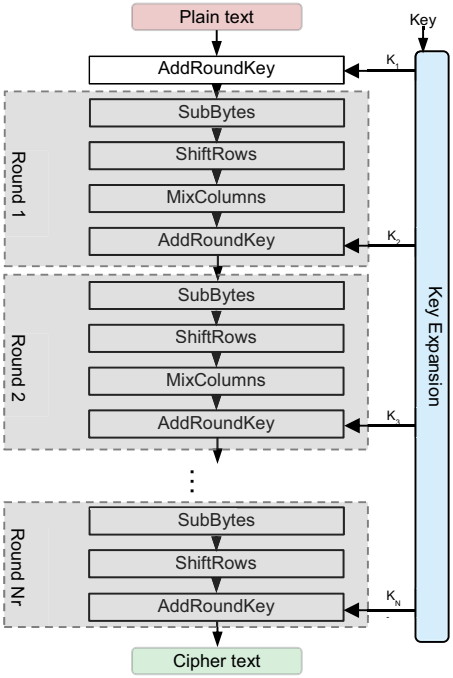

Figure 1: AES Algorithm

B. *Implementation of Unmasked S-box in Galios Field*

S-box of the "*Substitute Byte*" step is sometime implemented with look-up-table (LUT). However, implementation in combinational logic base on the Galois Field has reduced area in ASIC. It also has improved resistance to side channel attacks compared to the LUT implementation. The Galois Field implementation of the S-box involves taking the multiplicative inverse in $GF(2^8)$ followed by an affine transformation. However, this approach will cost lots of hardware resources. An alternative to this is to decompose the $GF(2^8)$ multiplicative inversion to lower order fields such as $GF(2^4)$, $GF(2^2)$, and GF(2) and convert back to $GF(2^8)$. A previously published paper has shown an efficient design based on this transformation [7]. Here we repeat some of the derivation to make the paper complete.

In Galois Field arithmetic, any arbitrary polynomial can be represented as $bx+c$, given an irreducible polynomial of x^2+Ax+B. Thus, the multiplicative inverse can be found with the following:

$$(bx+c)^{-1} = b(b^2B+bcA+c^2)^{-1}x+(c+bA)\,(b^2B+bcA+c^2)^{-1} \quad (1)$$

If the irreducible polynomial is $x^2+x+\lambda$, then the multiplicative inverse can be simplified to:

$$(bx+c)^{-1} = b(b^2\,\lambda\,+c(b+c))^{-1}x+(c+bA)\,(b^2\,\lambda\,+c(b+c))^{-1} \quad (2)$$

Converting this to Galois Field operations in hardware, we can obtain the following diagram for the multiplicative inversion module as shown in Figure 2. δ and δ^{-1} represent isomorphic and inverse isomorphic mappings to composite fields, and X represents a multiplication operation in $GF(2^4)$.

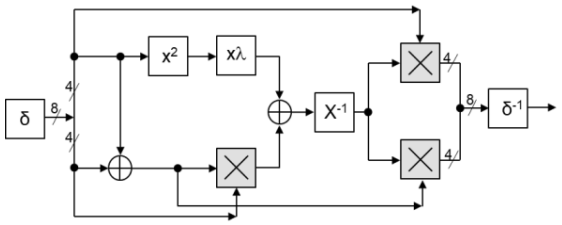

Figure 2: Multiplicative inversion module for the S-Box

III. AES MASKING IMPLEMENTATION

AES encryption consists of both linear and nonlinear transformations. Nonlinear transformations complicate the masking process. The idea behind masking is that before data enters a function, it must be added to a random mask. By this masking process the actual data values is hidden from any attackers. To unmask, we simply add the masked output to the transformed mask. Transformations such as *ShiftRows*, *MixColumns*, and *AddRoundKey* are linear operations. Masking and unmasking processes are relatively straight forward. *SubBytes* is a nonlinear operation and will require additional effort to obtain the transformed mask. In this section we cover the design of the masked AES.

A. *Masked AES Architecture and Core Design*

The AES implementation consists of the masked AES core and a 128 bit LFSR to generate the encryption masks. The masked AES core performs 128 bit encryption. The process is done in 10 cycles, computing 1 round per cycle, with the hardware of each round being reused to save area verses a fully unrolled implementation. The proposed masked AES is shown in Figure 3 where the original data (plaintext) is first masked by a random mask. The masked plaintext and the mask are, then, fed through the "masked AES core" which encrypt the masked data with the secret key. Result masked cipher-text is input into the unmasking module to arrive at the intended cipher-text.

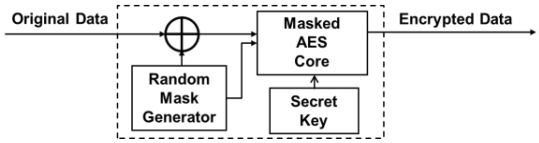

Figure 3: Masked AES Architecture

Figure 4 illustrates the zoom in block diagram of the masked AES core. Each of the layers in a round in the AES design is transformed into a masked design. For linear functions the design is relatively straight forward. The difficult function is the SubByte layer which is non-linear.

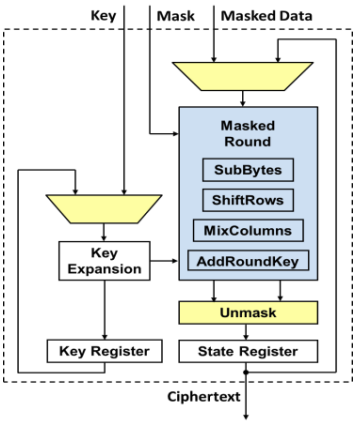

Figure 4: High Level Block Diagram of the Masked AES Core

B. Mask Generation

The mask used in each encryption round should be different to avoid the risk of being counteracted. A proper design should include a physical random number generator. For the demonstration purpose (without loss of generality) we used a 128 bit LFSR with taps at bits 128, 127, 126, 121 to generate fresh random mask at each round.

C. Masking Linear Transformations

To find the transformed mask for linear operations, it is simply done by passing the original mask that the data is masked with, through the same transformation. The reason behind this is illustrated in equation 3 where *f(a+m)* represents the masked data *(a+m)* transformed by a linear transformation *f*. The original output f(a) can be obtained by adding the transformed mask f(m), since any value XOR'd by the same number twice is the still the same value.

$$f(a+m) = f(a) + f(m) \quad (3)$$

D. Masking Non-Linear Trnasformations

The only nonlinear transformation in AES encryption is the *S-box*, and the root of the nonlinearity comes from the usage of AND operations mainly found in multipliers. From [6], masked AND can be implemented with the majority function MAJ.

$$MAJ(A, B, C) = AB + BC + AC \quad (4)$$

Substituting A with (A ⊕ M), B with (B ⊕ M), and C with M, we can simplify the expression to (A & B) ⊕ M, which is the equivalent to a masked AND operation. However, this setup requires that both inputs are masked with the same mask, thus a slight modification is needed to allow different masks to be used on the inputs as shown in Figure 5 below. The transformed mask is obtained from the XOR of A's mask and B's mask. All AND operations in the S-box are replaced with this masked AND.

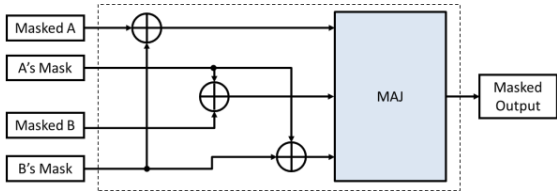

Figure 5: Masked AND Gate

IV. Results and Comparison

Both the unmasked and masked AES designs are synthesized and place and routed using TSMC's 22nm technology. The unmasked AES core uses 6754 cells and fits in a chip area of 100um x 100um. The masked AES uses 16107 cells and fits in a chip area of 130um x 130um which includes an additional 128-bit LFSR. This translates to roughly a 1.69x increase in chip area. Both designs operate at a clock frequency of 400MHz and reaches a throughput of 5.12 Gbps which shows that the masking method poses no performance loss. A comparison of performance, power and area between different side channel attack countermeasures is shown in Table 1 in the next page.

It is worth to note that the area of this design is scalable and it only relies on cells in the standard cell library. Design in [11] has a custom analog element which is not easily scalable. It also needs to custom build the needed cell from scratch. Moreover, this design also employs pipelining within a round. There are four pipe stages each for a function. This will enable higher throughput at the cost of slightly more area and energy. We believe, for IoT applications, the throughput of the design described in this paper achieved is sufficient for most cases.

To evaluate the SCA resistance of the masking scheme designed, the unmasked and masked AES are both implemented in TSMC's 22nm technology. Both designs are fully synthesized and place and routed. Post-layout netlists with RC annotation are simulated with CustomSim. Correlation Power Analysis (CPA) is performed on both designs using the Hamming weight power model. The chosen attack point is the first *SubBytes* operation since every 8 bits of the *SubBytes* output is determined only by the same 8 bits of the input. The unmasked AES is successfully attacked as shown in Figure 6. The peak in correlation corresponds to the correct key prediction.

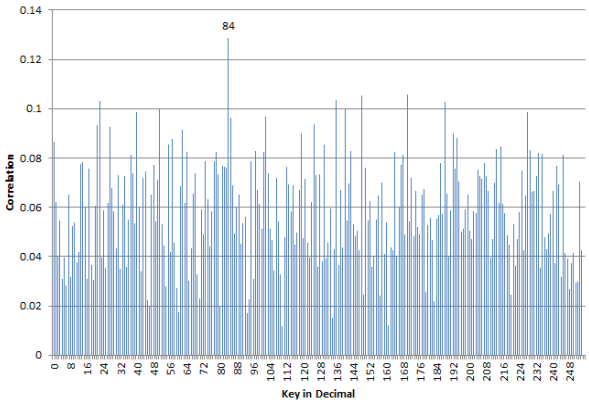

Figure 6: CPA on Unmasked AES

Clearly the sub-key value of "84" is identified through correlation analysis as marked. In contrast, the CPA results of the masked design showed no peak at value "84". Thus, the correct key is hidden from the attacker as shown in Figure 7.

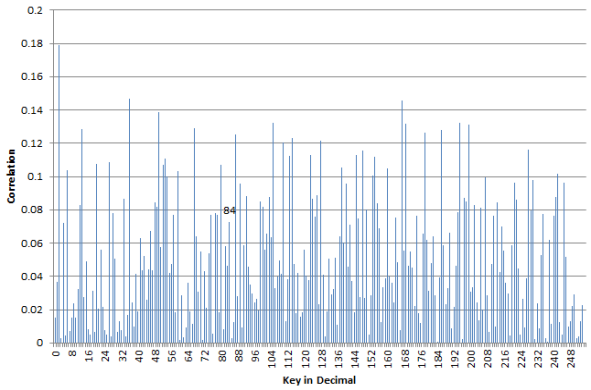

Figure 7: CPA on Masked AES

V. POSSIBLE IMPROVEMENT

The masked design has an area which is around 1.7X of the unmasked design. One way to reduce the area would be to reuse hardware in each round to compute the mask transformation. Since the majority of the AES operations are linear, the transformation hardware is the same as the round hardware. Depending on the target throughput, by passing the mask through the encryption round, it should be able to reduce the chip area by half at the cost of doubling the latency cycles (thus reducing the throughput by half) needed for the encryption.

VI. CONCLUSION

A high performance masked 128-bit AES engine has been implemented in TSMC 22nm technology. The encryption core runs at 400 MHz and has a throughput of 5.12 Gbps. The total chip area including a 128-bit LFSR is 130um x 130um, which is roughly a 1.69X increase compared to the original unmasked design. The masked design is verified by simulating the post APR design with CustomSim SPICE simulator and using Correlation Power Analysis (CPA). The results show that the masking scheme effectively hides the secret key.

REFERENCES

[1] "Announcing the ADVANCED ENCRYPTION STANDARD (AES)" Federal Information Processing Standards Publication 197. United States National Institute of Standards and Technology (NIST). November 26, 2001

[2] Paul Kocher, "Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems". Advances in Cryptology—CRYPTO'96. Lecture Notes in Computer Science. 1109: 104–113

[3] Paul Kocher, Joshua Jaffe, and Benjamin Jun, "Differential Power Analysis," Crypto 99 Proceedings, Lecture Notes in Computer Science Vol. 1666, M. Wiener, ed., Springer-Verlag, 1999

[4] Paul Kocher et al. "Introduction to differential power analysis ", J Cryptogr Eng (2011) 1: 5. https://doi.org/10.1007/s13389-011-0006-y

[5] Blömer J., Guajardo J., Krummel V. (2004) Provably Secure Masking of AES. In: Handschuh H., Hasan M.A. (eds) Selected Areas in Cryptography. SAC 2004. Lecture Notes in Computer Science, vol 3357. Springer, Berlin, Heidelberg

[6] W. Wei et al., "A compact implementation of masked AES S-box," 2012 IEEE 11th International Conference on Solid-State and Integrated Circuit Technology, Xi'an, 2012, pp. 1-3

[7] Edwin NC Mui "Practical Implementation of Rijndael S-Box Using Combinational Logic" 2007

[8] C. Tokunaga and D. Blaauw, "Secure AES engine with a local switched-capacitor current equalizer," 2009 IEEE International Solid-State Circuits Conference - Digest of Technical Papers, San Francisco, CA, 2009, pp. 64-65

[9] M. Doulcier-Verdier, J. Dutertre, J. Fournier, J. Rigaud, B. Robisson and A. Tria, "A side-channel and fault-attack resistant AES circuit working on duplicated complemented values," 2011 IEEE International Solid-State Circuits Conference, San Francisco, CA, 2011, pp. 274-276

[10] Y. Peng, H. Zhao, X. Sun and C. Sun, "A Side-Channel Attack Resistant AES with 500Mbps, 1.92pJ/Bit PVT Variation Tolerant True Random Number Generator," 2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), Bochum, 2017, pp. 249-254

[11] S. Lu, Z. Zhang and M. Papaefthymiou, "1.32GHz high-throughput charge-recovery AES core with resistance to DPA attacks," 2015 Symposium on VLSI Circuits (VLSI Circuits), Kyoto, 2015, pp. C246-C247

[12] Henrik Fegran, "DPA-Resistant ASIC Implementation of AES," MS Thesis, Norwegian University of Science and Technology, June 2015

TABLE 1. Power/Performance/Area Comparison Table

	[8]	[9]	[10]	[11]	[12]	This Work
Technology (nm)	130	130	130	65	65	22
Frequency (MHz)	110	50	100	1320	400	400
Throughput (Gb/s)	1.28	-	12.8	16.9	1.32	5.12
Unprotected Area (mm^2)	1.28	16500 Gates	-	0.097	-	0.01
Protected Area (mm^2)	1.37	27400 Gates	183.29K Gates	0.291	53K Gates	0.0169[+]
Power (mW)	44.34	-	-	98.0	167.9	41.6
Energy per block (nJ)	-	-	-	2.45[*]	17.6[*]	1.25[*]

[+] 16K Gates
[*] Calculated from throughput and power