Multiplicative Masked M&M: An Attempt at Combined Countermeasures with Reduced Randomness

Kaiyuan Li*, Haruka Hirata*, Daiki Miyahara*, Kazuo Sakiyama*, Yuko Hara-Azumi†, Yang Li*
*Department of Informatics, The University of Electro-Communications, Tokyo, Japan
{kaiyuan.li, h.haruka, miyahara, sakiyama, liyang}@uec.ac.jp
†Department of Information and Communications Engineering, Tokyo Institute of Technology, Tokyo, Japan
hara@cad.ict.e.titech.ac.jp

***Abstract*—With the advancement of hardware security, combined attacks with techniques such as side-channel analysis (SCA) and fault analysis (FA) have prompted the development of combined countermeasures. However, these countermeasures often come with significant system overhead. In this paper, we explore a potential multiplicative masking solution aimed at reducing the overhead while maintaining security claims. We demonstrate the approach with Mask & Macs (M&M), a scheme that combines Boolean masking and MAC tag redundancy to provide SCA and DFA protection, addressing the challenge of its substantial overhead, particularly the high randomness requirement. We introduce a novel multiplicative masking scheme as a replacement for conventional threshold implementation (TI) modules partially, leading to a reduction of over 50% in randomness requirement. While our initial results show promise in reducing overhead, other limitations remain, and further research is needed to address other challenges. This work provides a new perspective on improving combined countermeasures by exploring ways to reduce system overhead.**

***Index Terms*—multiplicative masking, Kronecker delta function, combined countermeasures, Masks & Macs, S-Box, AES.**

I. Introduction

Security always comes with a cost, and the same applies in the field of hardware security. Since hardware attacks were introduced in the 1990s, side-channel analysis (SCA) and fault analysis (FA) have become two major types of hardware attacks. In contrast, masking and detection are responsive countermeasures against SCA and FA. If we look back into the history of hardware security, we can see that both attacks and countermeasures have been developed with increasing complexity, following stricter prerequisites. As a result, keeping the balance between security claims and implementation overhead is becoming a challenging subject for researchers.

For example, Masks & Macs (M&M) [1] is a cryptographic scheme that uses masking and redundancy against SCA and differential fault analysis (DFA) [2]. However, due to the use of redundancy, M&M requires double the resource overhead than a conventional threshold implementation [3] AES (TI-AES), especially hundreds of fresh random bits per cycle, which is a disadvantage for IoT applications. We found that integrating a novel multiplicative masking scheme with a Kronecker delta function [4] into its AES S-Boxes can significantly reduce the randomness requirement of the system by 58.0% at second-order, as well as comparable resource overhead, latency, and security claims. This also means potential optimization of area overhead with a more compact PRNG in the future.

A. Related Works

SCA is a type of attack that utilizes leakage from unexpected channels that cipher designers cannot anticipate. The most used side-channel information is power consumption [5], [6]. With a proper analysis model, an attacker can dig the secret key through a divide-and-conquer approach. In contrast, masking uses randomness to decompose the secret value, making the attacker's observation more difficult. Masking can be implemented using arithmetic, Boolean, multiplicative operations, etc. Meanwhile, masking can be security provable against different adversarial models, such as TI [3], and domain-oriented masking (DOM) [7].

On the other hand, fault analysis induces abnormal system behaviors by deliberately injecting faults into the computation process. These behaviors can vary with the intermediates of the computation, thus revealing the secret value. There are various types of fault analysis: DFA, which utilizes the differential characteristics of faulty outputs; fault sensitivity analysis (FSA) [8], which utilizes the relationship between intermediate values and the probability of fault occurrence; and ineffective fault analysis (IFA) [9], in which faults do not affect the outputs. To protect against fault attacks, it is common to utilize time or area redundancy to perform additional detection, terminating the computation timely, or randomizing the output if a fault is detected.

As the research on hardware security becomes more profound and systematic, combined countermeasures that can protect the system against both SCA and FA are gradually coming into our sight. For example, ParTI [10] combines masking and redundancy simultaneously to protect against both types of attacks. However, IFA that threatens cryptographic systems by injecting undetectable faults can seriously compromise security claims. CAPA [11] is another combined countermeasure based on multi-party computation (MPC).

The final published version is available at IEEE Xplore: https://doi.org/10.1109/TrustCom63139.2024.00112

Although it incurs high overhead, it provides very strong security guarantees. However, it was also recently broken by CAPABARA [12], a new combined attack.

B. Contribution

For the first time, we applied multiplicative masking to a MAC tag redundancy combined countermeasure (i.e., the M&M), demonstrating its adaption for the information-theoretic MAC tag approach. Our first- and second-order multiplicative masked S-Boxes, along with a complete second-order multiplicative masked M&M-AES, yield comparable FPGA resource utilization and latency to the original design. Notably, the complete second-order implementation achieved a significant 58.0% reduction in randomness requirements, indicating the potential for embedding a more compact PRNG to further optimize the area overhead. Furthermore, our new multiplicative masked M&M passes a practical SCA evaluation, while maintaining its security claims against DFA. While the limitations within the M&M remain and need further perfection, we are offering an alternative path to construct more cost-effective combined countermeasures.

The paper is structured as follows: Section II provides knowledge on M&M and multiplicative masking. Section III outlines the methodology for applying multiplicative masking to the M&M scheme. Section IV presents the implementation details of the proposal. Section V shows the implementation results. Section VI provides both theoretical and practical security analyses we have done so far. Section VII concludes the paper and indicates future research directions.

II. Preliminaries

A. Mask & Macs (M&M)

M&M is a combined countermeasure that includes masking, redundancy, and infective computation. As we know, the masking technique protects the system from SCA. On the other hand, redundancy and infective computation prevent the system from FA, such as the commonly used differential fault analysis (DFA), by detecting faults and infecting (randomizing) the output.

The redundancy is realized by information-theoretic MAC tags. As shown in Figure 1, the M&M is a dual-rail system. The input pair includes a plaintext p and its corresponding tag $\tau^p = \alpha p$ constructed with a secret α. They are then independently processed by the same cryptographic algorithm (e.g., AES). The cipher for the tag is slightly modified to ensure that the α times relation holds after encryption. If the relation holds after the encryption (i.e., $\alpha c = \tau^c$), the system will output the correct ciphertext. Otherwise, the output will be infected unbiasedly, preventing potential attackers from gathering useful information.

B. Multiplicative Masking

Multiplicative masking was first proposed by [13] in 2001. However, it is less popular than Boolean masking schemes due to its critical vulnerability: it cannot mask a zero input. Consider a data byte p masked by a random byte r, resulting in rp. When p equals zero, the rp will also remains zero unmasked, so that an attacker can easily recognize it and perform a DPA attack [14].

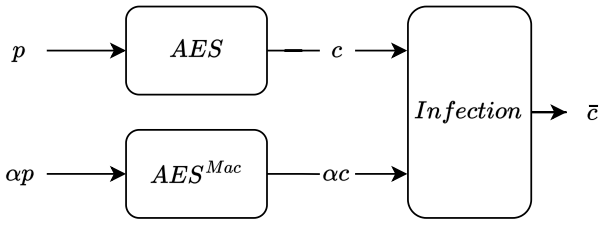

Fig. 1. Overview of the M&M-AES scheme.

Nevertheless, when performing $GF(2^8)$ multiplicative inversion in AES, multiplicative masking is better suited than Boolean masking. The operation can be done locally on rp rather than all the shares: $(rp)^{-1} = r^{-1}p^{-1}$. To recover p^{-1}, we only need to multiply it by r. Meanwhile, Boolean masking is more appropriate for linear operations, making each scheme valuable for different scenarios.

A glitch-resistant software implementation that remaps zero to another element was proposed in 2011 [15] to address the zero problem. However, hardware implementations are more challenging due to resource constraints and lack of resiliency, limiting the study of multiplicative masking schemes.

In 2018, De Meyer et al. [4] proposed a new hardware-oriented approach based on the earlier software implementations addressing the zero problem. The key idea is to use the Kronecker delta function to detect zero inputs before they enter the S-Box, substituting them with one. Since multiplicative inversion in $GF(2^8)$ is bijective and the inverses of zero and one are themselves, the value remains unchanged after the inversion and can be recovered to zero before the affine transformation. The multiplicative inverse is calculated using a minimal unshared circuit. The scheme converts between Boolean and multiplicative masking before and after inversion to maximize efficiency.

III. Methodology

A. Notations

To implement the the methodology from [4] to the M&M, we now standardize the notations from both schemes for more intuitive comprehension. We denote multiplication and addition in $GF(2^8)$ as "$\otimes$" and "$\oplus$". We use bold font to indicate shared values, such as $\boldsymbol{x} = (b_0, b_1, \ldots, b_d)$.

Additionally, matrices from $GF(2)^{8\times 8}$, vectors from $GF(2)^8$ as well as the computations between them are denoted with boldface font. The isomorphism between $GF(2^8)$ and $GF(2)^8$ is denoted with ϕ, such as $\phi(\alpha x) = \mathbf{M}_\alpha \mathbf{x}$.

B. S-Box

A simplified structure of multiplicative masked AES S-Box without zero issue solutions is shown in Figure 2. For simplicity, we will demonstrate a $d = 2$ scenario including 3 shares. However, the masking proposal can be expanded to arbitrary d^{th}-order.

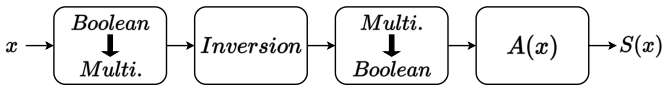

Fig. 2. Overview of the multiplicative masked S-Box.

Boolean to Multiplicative. Consider a Boolean shared value x, which is $\boldsymbol{x} = (b_0, b_1, b_2)$. We introduce a non-zero random multiplicative mask r_0 from $GF(2^8)$ and multiply all Boolean shares by it. We obtain 4 shares:

$$\boldsymbol{x} = (r_0, b_0', b_1', b_2'),\ b_i' = b_i \otimes r_0 \text{ for } i = 0, 1, 2.$$

This is called the expansion. To recover $d + 1 = 3$ shares, we cancel the first Boolean share b_i' on the left (which is b_0' currently), by recombining it to its next share. This is called the compression:

$$\boldsymbol{x} = (r_0, b_1'', b_2'),\ \text{where } b_1'' = b_1' \oplus b_0'.$$

The first iteration comes to an end. We can see that the b_0 is not simply discarded but recombined to b_1, since

$$b_1'' = b_0' \oplus b_1' = (r_0 \otimes b_0) \oplus (r_0 \otimes b_1) = r_0 \otimes (b_0 \oplus b_1). \quad (1)$$

For the second iteration, draw another non-zero random r_1, multiply all the b_is by it:

$$\boldsymbol{x} = (r_0, r_1, r_1 \otimes b_1'', r_1 \otimes b_2').$$

Then we recombine the first Boolean share on the left (which becomes b_1'' now) to its next share. We obtain

$$b_2'' = (r_0 \otimes r_1) \otimes (b_0 \oplus b_1 \oplus b_2) = (r_0 \otimes r_1) \otimes x,$$

and the sharing can be also written as

$$\boldsymbol{x} = (r_0, r_1, b_2''),\ \text{where } b_2'' = (\bigotimes_{i=0}^{1} r_i) \otimes x. \quad (2)$$

In general, for arbitrary d^{th}-order Boolean sharing, the mask switching from Boolean to multiplicative can be done after d iterations with d non-zero random numbers:

$$\boldsymbol{x} = (r_0, r_1, \ldots, r_{d-1}, b_d''),\ \text{where } b_d'' = (\bigotimes_{i=0}^{d-1} r_i) \otimes x.$$

Inversion. The value x is now stored in the last share b_2'' as (2) shows. Naturally, the calculation of x^{-1} can be done on b_2'' locally:

$$\boldsymbol{x^{-1}} = (r_0, r_1, b_2^{''-1}),\ \text{where } b_2^{''-1} = (\bigotimes_{i=0}^{1} r_i^{-1}) \otimes x^{-1}, \quad (3)$$

in which the r_i^{-1}s can be cancelled by multiplying the r_is. The expression for arbitrary d^{th}-order is

$$b_d^{''-1} = (\bigotimes_{i=0}^{d-1} r_i^{-1}) \otimes x^{-1}.$$

Multiplicative to Boolean. The shared $\boldsymbol{x^{-1}}$ is represented by (3). To switch the masking to Boolean securely, we firstly expand the sharing by introducing a random Boolean mask r_2 noted as $b_1^{'-1}$, adding it to the last share $b_2^{''-1}$:

$$\boldsymbol{x^{-1}} = (r_0, r_1, b_1^{'-1}, b_2^{'-1}),\ \text{where } b_2^{'-1} = b_1^{'-1} \oplus b_2^{''-1}.$$

Then we remove the last multiplicative share (which is r_1 currently) by multiplying all Boolean shares by it:

$$\boldsymbol{x^{-1}} = (r_0, b_1^{-1}, b_2^{-1}),\ \text{where } b_i^{-1} = r_1 \otimes b_i^{'-1} \text{ for } i = 1, 2.$$

The first iteration comes to an end. $b_2^{''-1}$ is now split into two shares by Boolean masking and r_1 is canceled (which can be proved with (3)). Repeat the steps above with another Boolean mask r_4 (noted as b_0^{-1}) for the second iteration, then

$$\boldsymbol{x^{-1}} = (b_0^{-1}, b_1^{-1}, b_2^{-1}) \quad \text{where} \quad x^{-1} = \bigoplus_{i=0}^{2} b_i^{-1}.$$

The sharing is now Boolean masked for linear operations afterward. For arbitrary d^{th}-order masking scenario, the switching can be done after d iterations with d random numbers:

$$\boldsymbol{x^{-1}} = (b_0^{-1}, b_1^{-1}, \ldots, b_d^{-1}) \quad \text{where} \quad x^{-1} = \bigoplus_{i=0}^{d} b_i^{-1}.$$

Register Requirement. We need to insert registers between the expansion and compression phases to prevent glitches. Consider the expansion phase in (1):

$$b_1'' = b_0' \oplus b_1' = [r_0 \otimes b_0] \oplus [r_0 \otimes b_1] = r_0 \otimes (b_0 \oplus b_1).$$

There is a chance that r_0 arrives at the multipliers later than b_0 and b_1 as the latter form of the result shows. This means without inserting a register stage, two shares of x are now recombined at the adder of the compression phase and the security is reduced by one order. Registers here, denoted by square brackets, will prevent the risk of recombination. More generally, for a d^{th}-order sharing, we need $2d$ register stages to prevent glitches.

Affine Transformation. The affine transformation of the AES S-Box is a linear operation and can be expressed as $A(x^{-1}) = l(x^{-1}) \oplus 0\text{x}63$. Nevertheless, the α times relation between plaintext and its tag in the M&M will not hold naturally within the S-Box for an AES round.

For the MAC tag $\tau^x = \alpha x$ corresponding to an x, its multiplicative inverse is $(\tau^x)^{-1} = \alpha^{-1}x^{-1}$, but not the valid tag $\tau^{(x^{-1})} = \alpha x^{-1}$ for the x^{-1}. So it is necessary to multiply $(\tau^x)^{-1}$ by α^2. This includes shared $\boldsymbol{\alpha}$ squaring and shared multiplication of $\boldsymbol{(\tau^x)^{-1}} \otimes \boldsymbol{\alpha^2}$ in the actual implementation.

However, it can be further optimized by modifying the affine transformation with the isomorphism between $GF(2^8)$ and $GF(2)^8$. For an $\alpha \in GF(2^8)$, there always exist a corresponding matrix $\mathbf{M}_\alpha \in GF(2)^{8\times 8}$ so that $\mathbf{M}_\alpha \mathbf{x} = \phi(\alpha x)$. The same is true for the $l(x)$ in the affine transformation $A(x)$:

$$\exists \mathbf{M_L} \in GF(2)^{8\times 8} \quad \text{s.t. } \mathbf{M_L x} = \phi(l(x)).$$

Then the calculation from $(\tau^x)^{-1} = \alpha^{-1}x^{-1}$ to $\tau^{l(x^{-1})} = \alpha l(x^{-1})$ in $GF(2^8)$ can be merged into:

$$\phi(\tau^{l(x^{-1})}) = \mathbf{M}_\alpha \mathbf{M_L} \mathbf{M}_\alpha \phi((\tau^x)^{-1}). \quad (4)$$

Now we can pre-compute $\mathbf{M}_\alpha \mathbf{M_L} \mathbf{M}_\alpha$ in shares and perform shared multiplication on shared $\boldsymbol{(\tau^x)^{-1}}$ in one step.

C. Addressing Zero Issues of Multiplicative Masking

Zero Inputs. A zero input compromises the security of the multiplicative masking, making the scheme vulnerable to DPA [14]. As a solution, we can map the input 0 to 1 in $GF(2^8)$ by a $\delta(x)$ function before the inversion and recover it to 0 afterward. This has been applied to software-oriented multiplicative masking schemes such as [15] and has been proved secure against SCA. We also apply the Kronecker delta function in multiplicative masking:

$$\delta(x) = \begin{cases} 1 & x = 0 \\ 0 & x \neq 0 \end{cases}$$

A simple solution for $x \in GF(2^8)$ could be $\delta(x) = \overline{x_0}\&\overline{x_1}\ldots\&\overline{x_7}$. Then the multiplicative inversion in $GF(2^8)$ can be written as

$$x^{-1} = (x \oplus \delta(x))^{-1} \oplus \delta(x). \quad (5)$$

The 0 inputs will be mapped to 1 by the $\delta(x)$, and then recovered to 0 after the inversion. Other values will not be affected. However, the value x is a shared input, which means $\delta(x)$ should be a shared operation as well.

Zero Randomness. Any multiplicative mask cannot be zero either, since the product of rp will become zero and lose p permanently. However, to the best of our knowledge, there is no straightforward way to securely generate uniformly distributed non-zero randomness with low overhead. [4] suggested using FIFOs to buffer the non-zero randomness for multiplication (i.e., r_0 in the first order and r_0, r_1 in the second order) and resupply them when the S-Box is idle. It makes sense that the M&M-AES is implemented with a byte-serialized structure [16], which means for an S-Box with a latency of C cycles ($C \geq 4$), a round will be $16 + C$ cycles, in which S-Box is only active for 20 cycles. As a result, we can resupply the FIFOs during the idle cycles, preventing zero randomness from entering the S-Box.

However, such a solution does not solve the issue completely and will eventually cause malfunctions with high probability. We will further discuss it in the next section.

IV. Multiplicative Masked M&M-AES Integration

A. S-Box

Following the methodology described in Section III-B, we construct first- and second-order S-Boxes for the M&M-AES.

First-Order Inversion. The first-order inversion featuring mask switching is shown in Figure 3. The left part of the figure converts a Boolean sharing $\boldsymbol{x} = (b_0, b_1)$ to a multiplicative sharing $\boldsymbol{x} = (r_0, b_1'')$, while the right part performs the reverse conversion. Recall that we have to insert registers between

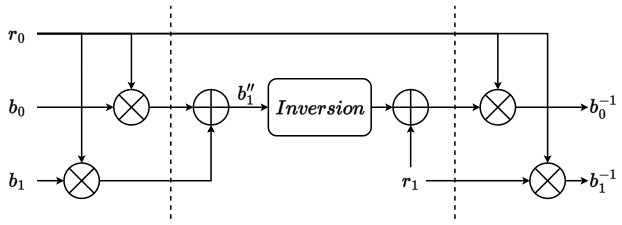

Fig. 3. First-order shared inversion [4]. Dashed lines indicate register stages.

phases to prevent glitches, resulting in two register stages in the circuit. As for the inversion, since it can be done on b_1'' locally, a modified version of the compact unshared inversion circuit from [17] is used as in [4]. There are two 8-bit randomness used in total.

Second-Order Inversion. The second-order implementation performs an additional iteration as shown in Figure 4. Nevertheless, this approach does not guarantee the second-order security [4]. Consider a pair of intermediates

$$s_1 = (b_0 \otimes r_0) \oplus (b_1 \otimes r_0)$$
$$s_2 = b_2$$

around the first register stage in Figure 4. Since r_0 is non-zero, the probability of s_1 being zero depends on $b_0 \oplus b_1$, which occurs with a probability of 2^{-8}. If true, the value x is equal to b_2 because $x = b_0 \oplus b_1 \oplus b_2$. This equation shows that s_2 leaks information of x under the circumstance. An attacker may construct a leakage model based on Hamming weight behaviors [4]. A similar risk of leakage exists in the multiplicative to Boolean conversion as well.

As a solution, an extra random u is introduced [4], masking the shares at these vital timings. By recycling u in mask switching modules (4 bits for each module on average), the inversion can be done with five 8-bit randomness in 4 cycles.

Affine Transformation. Affine transformation for the MAC tag is a shared operation as described in Section III-B. We apply the same approach from [1], making use of the isomorphism between $GF(2^8)$ and $GF(2)^8$ as shown in (4). The control logic generates shared $\mathbf{M}_\alpha$ and passes it to the $\boldsymbol{A(x)}$ module at the start of every encryption.

The shared affine transformation consumes 1 cycle as well as 8/24 bits for first/second-order implementation. Note that the $\boldsymbol{A(x)}$ for the plaintext can be done locally, but still has to wait for 1 cycle to synchronize with its MAC tag.

B. Zero Issue Solutions

Kronecker Delta Function. The Kronecker delta function uses 1-bit DOM-*indep* multipliers [7] as shared AND gates to build the circuit. To balance the trade-off between latency and area cost, a 4-2-1 fan-out pattern is chosen as shown in Figure 5. A DOM-AND gate requires one cycle to calculate, as well as 1/3 bits per gate for first/second-order.

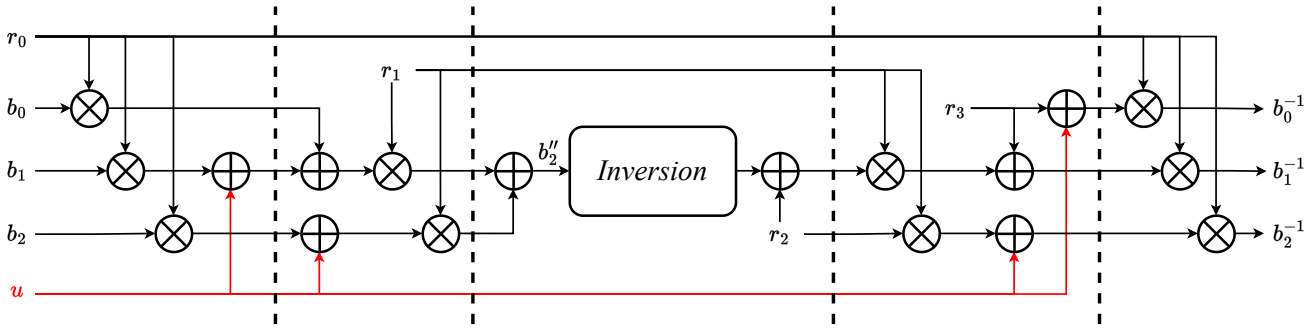

Fig. 4. Second-order shared inversion [4]. Dashed lines indicate register stages.

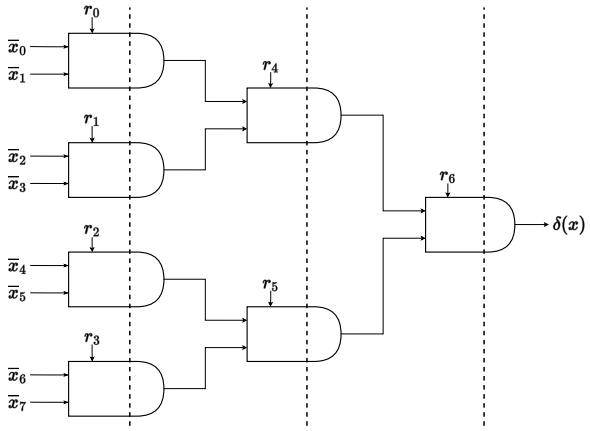

Fig. 5. Shared Kronecker $\delta(x)$. Dashed lines indicate register stages.

Note that the input value x is shared and needs to be flipped. The flipping methods vary for different orders:

$$\begin{aligned} \text{1st}: &\quad \overline{x} = \overline{b_0} \oplus b_1 = b_0 \oplus \overline{b_1}\,; \\ \text{2nd}: &\quad \overline{x} = \overline{b_0} \oplus \overline{b_1} \oplus \overline{b_2}. \end{aligned}$$

In total, the $\delta(x)$ requires $(4+2+1)\times 1 = 7$ bits for first-order, and $(4+2+1)\times 3 = 21$ bits for second-order implementation. The latency of both is 3 cycles.

FIFOs for Randomness. Despite the FIFOs described in Section III-C, the possibility of malfunction persists. Consider a first-order S-Box with a depth-of-2 FIFO fully resupplied at cycle 1 in a round. The probability of a proper functioning round is (n represents the number of zeros generated by the PRNG in a round):

$$P_{Rd} = \sum_{i=0}^{2} P(n=i) \approx 99.994\%.$$

Then the probability for a correct encryption in the M&M-AES (2 S-Boxes for 10 rounds) is $P_{Enc} = ((P_{Rd})^2)^{10} \approx 99.880\%$. The probability will decrease rapidly with hundreds or thousands of encryption, which is extremely possible in real-world usage. Responsively, our $RandFIFO$s implement the following features:

1) Initialize with non-zero values
2) Repeat its last output when empty
3) Start enqueuing randomness before encryption begins
4) Hold and resupply during idle cycles

The first two features will absolutely ensure that no zero randomness will be used as multiplicative masks, while the latter two help provide high-quality randomness unbiasedly. We assign one $RandFIFO$ to each multiplication mask input port. While not optimal, the $RandFIFO$s have shown no distinguishable security degradation in our evaluations so far.

C. Overview

Table I summarizes the latency and randomness overhead of the modules, and Figure 6 depicts an S-Box featuring zero-issue solutions stated earlier. Note that there are 2 S-Boxes in M&M-AES to process the plaintext and its MAC tag separately, resulting in doubled overhead for all modules except for affine transformation, since only the MAC tag needs shared multiplication.

With the byte-serialized structure [16], one round of our M&M-AES implementation will cost $16+6 = 22$ cycles for first-order and $16+8 = 24$ cycles for second-order as mentioned in Section III-C. With 20 S-Box-active cycles (16 cycles for *SubBytes* and 4 cycles for *SubWords*), there are 2 and 4 idle cycles to resupply the $RandFIFO$s. The idle cycles appear at the end of a round, thus we hold the output of $RandFIFO$s from cycle 20 to the end of every round.

V. Implementation Results

We implemented the multiplicative masked S-Boxes for both first- and second-order, and a complete second-order

TABLE I
Overhead Composition of M&M-AES S-Box Variants

Variants / *Modules*	First-Order Latency (cycles)	First-Order Randomness (bits/cycle)	Second-Order Latency (cycles)	Second-Order Randomness (bits/cycle)
Proposal	**6**	**54**	**8**	**146**
Kronecker $\delta(x)$	3	7×2	3	21×2
Bool. $\rightarrow$ *Multi.*	1	8×2	2	(16+4)×2
Unshared Inv.	-	-	-	-
Multi. $\rightarrow$ *Bool.*	1	8×2	2	(16+4)×2
Affine Trans.	1	8	1	24
M&M [1]	**6**	**116**	**6**	**348**
Shared Inv.	5	54×2	5	162×2
Affine Trans.	1	8	1	24

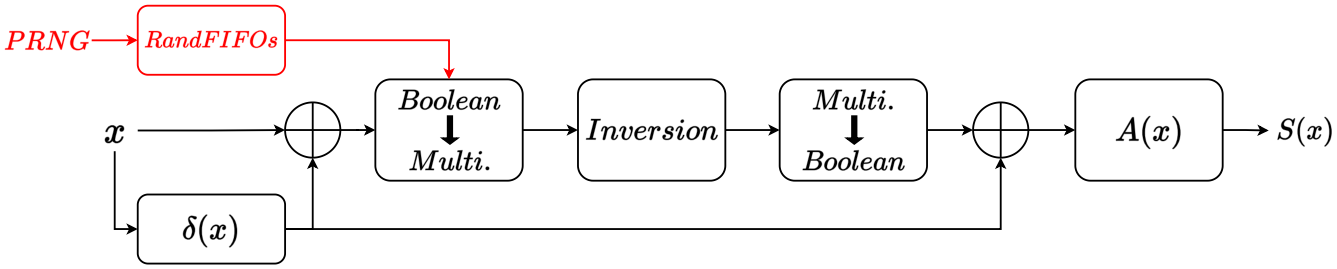

Fig. 6. Overview of the proposed multiplicative masked S-Box. The red datapath represents the non-zero randomness used as multiplicative masks.

M&M-AES. All the implementations were performed on Xilinx ISE 14.7 by VHDL with `Keep Hierarchy` option on. The target device was a Xilinx Spartan-6 FPGA.

Table II presents the implementation results of S-Box variants. Note that we do not possess the original first-order M&M project files, hence the first-order FPGA resource costs are not compared. In second-order implementations, the utilized FF count increases by 25.5%, while the LUT count decreases by 6.4%. As for randomness overhead, we confirmed a significant 53.4% reduction for first-order and 58.0% for second-order. Moreover, the multiplicative masked S-Boxes demonstrate comparable latency to those in the original M&M.

Table III compares the original second-order M&M-AES with our proposed scheme. Our implementation largely maintains the FPGA resource utilization, with a slightly higher latency. However, the random bits required per cycle are significantly reduced by 58.0%. It suggests the possibility of employing a more compact PRNG to optimize the FPGA resource (area) overhead even further.

TABLE II
IMPLEMENTATION RESULTS OF AES S-BOX VARIANTS

Order	S-Box Variants	Resources FFs	Resources LUTs	Latency (cycles)	Randomness (bits/cycles)
1st	M&M [1]	N/A	N/A	6	116
	Proposal	172	590	6	54
	Change	**N/A**	**N/A**	**0.0%**	**-53.4%**
2nd	M&M [1]	636	1956	6	348
	Proposal	798	1831	8	146
	Change	**+25.5%**	**-6.4%**	**+33.3%**	**-58.0%**

TABLE III
IMPLEMENTATION RESULTS OF SECOND-ORDER M&M-AES VARIANTS

M&M-AES Variants	Resources FFs	Resources LUTs	Latency (cycles) Round	Latency (cycles) Overall	Randomness (bits/cycle)
M&M [1]	3585	6418	22	239	348
Proposal	3748	6500	24	259	146
Change	**+4.5%**	**+1.3%**	**+9.1%**	**+8.4%**	**-58.0%**

VI. SECURITY EVALUATION

We have so far, verified the security of the proposal theoretically following similar approaches as per [1] and [4] at our best effort. Additionally, The practical SCA security is confirmed by a non-specific TLVA test. More theoretical and practical analyses still need to be done.

A. SCA Evaluation

Theoretical Analysis. The multiplicative masking scheme [4] we applied to the M&M-AES is designed against d-probing model. Especially, it is proved to be strong non-interference (SNI) [18] secure, whose adversarial model is stronger than the d-probing. Also, the authors used a software tool to verify the non-completeness of the S-Box and exhaustively probed every possible glitch to prove the glitch security [4] of the circuit. However, as the authors indicate, the SNI and glitch security that are proven separately do not guarantee composability in a real-world glitchy environment. Alternatively, the authors confirmed its SCA security by practical experiments.

Another concern is whether the $\delta(x)$ remapping can effectively circumvent the zero-input DPA attack targeting a conventional multiplicative masking scheme. The shared $\boldsymbol{\delta(x)}$ output is XORed to input value shares, only remapping zero to one and then recovering it to zero afterward as (5) shows. Thus, all the shared inputs including zero can be properly masked by multiplication for at least d^{th}-order at any time. We must note that it will slightly alter the joint distribution of input shares that $P(x = 1) = \frac{2}{256}$, leaving the probability of other values unchanged. However, in a practical implementation, we believe this faint bias can be largely hidden due to the noisy power consumption of other circuit modules.

Although there are still flaws in the proof of the multiplicative masking scheme, we verify its practical SCA security through experimental evaluation.

Experimental Evaluation. The experimental SCA evaluation setup is based on a SAKURA-G board, identical to that used in the M&M paper [1]. Our modified second-order M&M is programmed into the main Spartan 6 FPGA for encryption, while the auxiliary FPGA handles communication with the host PC. The FPGAs are clocked at 3.072 MHz.

We conducted a non-specific test vector leakage analysis (TVLA) as per [19]. Traces of the complete encryption process were collected at 500 MSa/s, with each trace containing about 42,000 sample points. This extends beyond previous analyses: the original M&M scheme [1] examined only 1.2/1.4 rounds for the first- and second-order, while the original multiplicative masking scheme [4] analyzed 2.5 rounds. TVLA performs Welch's t-test on two groups of traces: fixed plaintext versus

random plaintexts. The t-test checks if the null hypothesis holds at d^{th}-order. If the absolute value of t-statistics exceeds a threshold of 4.5, it indicates there is a distinguishable difference between fixed and random plaintext traces, i.e., leakage is confirmed with 99.9995% confidence at d^{th}-order.

The TVLA results for our multiplicative masked M&M-AES are shown in Figure 7. As we can see in the left column of the figure, PRNG-off (masking-off) test results exceed the threshold with merely 100 thousand traces at every order. In contrast, PRNG-on (masking-on) test results in the right column show no leakage with 50 million traces up to the second order, confirming our design's practical SCA security up to the second order.

B. FA Evaluation

Differential Fault Analysis (DFA). Our proposal keeps the MAC tag redundancy and infection mechanism from the original M&M [1] unchanged. We alternatively use "+" and "$\cdot$" to replace "$\oplus$" and "$\otimes$" for readability in the following. The infection can be written as

$$\bar{c} = (c + \Delta^c) + R \cdot (\alpha \cdot (c + \Delta^c) + \tau^c + \Delta^\tau),\ R \neq 0. \quad (6)$$

The Δ^c (resp. Δ^τ) represents faults propagated to the ciphertext (resp. ciphered tag). The key of a DFA attack is to obtain the faulty ciphertext $c+\Delta^c$, or Δ^c. We can simplify (6) knowing that $\tau^c = \alpha \cdot c$:

$$\bar{c} = c + (1 + R \cdot \alpha) \cdot \Delta^c + R \cdot \Delta^\tau.$$

The plaintext and its tag are ciphered in shares separately during the encryption. Thus, the probability that $\Delta^\tau = \alpha \cdot \Delta^c$ (i.e., obtain $c+\Delta^c$) is 2^{-k} for the adversary, which is no better than pure guessing. We now discuss the opposite scenario: $\Delta^\tau \neq \alpha \cdot \Delta^c$. We can verify that the $\bar{c}$ is uniformly random in $GF(2^8)$ if R is a uniformly drawn non-zero randomness.

If we further assume our adversary with stronger ability that he can know the R and inject a known fault into the α, we then replace the α in (6) with $\alpha + \Delta^\alpha$. Simplify the equation again and obtain

$$\bar{c} = c + (1 + R \cdot \alpha + R \cdot \Delta^\alpha) \cdot \Delta^c + R \cdot \Delta^\tau.$$

Still, the $\bar{c}$ is uniformly randomized even though almost all the variables are known to the adversary except the α, as long as it is shared and kept secret to the adversary.

Here we conclude the resistance against DFA: as long as the injected faults can propagate to c, τ^c or both, the proposed multiplicative masked M&M is able to detect the fault and infect the output unbiasedly with a probability of $1 - 2^k$.

Ineffective Fault Analysis (IFA). As the authors of the original M&M noted, IFA and SIFA fall outside the M&M's adversary model, since they are not detectable at the algorithm level [1]. Recently, a SIFA2-like attack targeting zero-input characteristics of the inversion in Carnight's S-Boxes from the M&M has been proposed [20]. It exploits an early-stage determination for zero inputs in the S-Boxes, succeeding in recovering all of the AES key bytes. Comparing to the Carnight's S-Boxes, the multiplicative masked S-Boxes in our scheme possess two properties: firstly, $\delta(x)$ ensures no zero input will enter the S-Boxes. Secondly, the S-Boxes' latency and behaviors remain consistent across all inputs. As a result, this SIFA2-like attack [20] does not apply to our scheme.

While our proposed scheme builds on the strengths of the M&M, it shares similar limitations in terms of protection against IFA/SIFA. A thorough glitch analysis in [4] shows extreme unlikelihood to perform a glitch attack on the multiplicative masked S-Box, but we consider a potential to perform IFA/SIFA attacks on the $\delta(x)$, as the inputs vary in the $GF(2^8)$ but the output is only zero or one, resulting in a huge entropy loss. Theoretically, scenarios may exist where the output of $\delta(x)$ remains unchanged despite bit-wise errors occurring, potentially enabling IFA/SIFA attacks. Fault-resist zero check mechanism is left as an open question for future work.

C. Combined Attacks Evaluation

The M&M provides security to many combined attacks, such as PACA, which requires faulty ciphertext to succeed [1]. Some other combined attacks targeting the checking mechanism or correlations on the faulty ciphertext are thwarted because of the improved infection with unbiased randomization [1]. Our modified M&M holds the same resistance according to the description in former subsections.

The M&M is a rather lightweight combined countermeasure that does not provide provable security with a firm combined security model like the multi-party computation (MPC) protocol does. However, the well-known MPC-based combined countermeasure CAPA [11], is also broken by a new combined attack named CAPABARA [12] recently.

Also, as our analysis on IFA families in the last subsection shows, since the ineffective faults will disappear during the fault propagation, the infection will not work, causing the success of a SIFA2-like attack [20]. In the same paper, Hirata et al. proposed a fine-grained λ-detection at the critical timings in the S-Boxes to record the faults before they vanish. Such a solution can indeed detect ineffective faults effectively but also bring a higher area overhead of 1.33 times compared to the M&M. The solution is an effective practical approach but still not provably secure.

To conclude the current progress of combined security, several provable protocols are proposed, such as NINA [21] and CINI-MINIS [22]. They provide provable security against SIFA, but the former lacks practicability in hardware implementation and the latter presents 4.40 times of area overhead comparing to the M&M. Thus, the continuous improvement on the M&M still remains a valuable direction to provide cost-effective combined security.

VII. Conclusion and Future Work

In this paper, we applied a novel multiplicative masking to the M&M scheme, confirming a significant reduction of over 50% in randomness cost while maintaining other overheads. Moreover, our design still holds SCA and FA security claims from the original M&M. This work may inspire research on more cost-effective combined countermeasures.

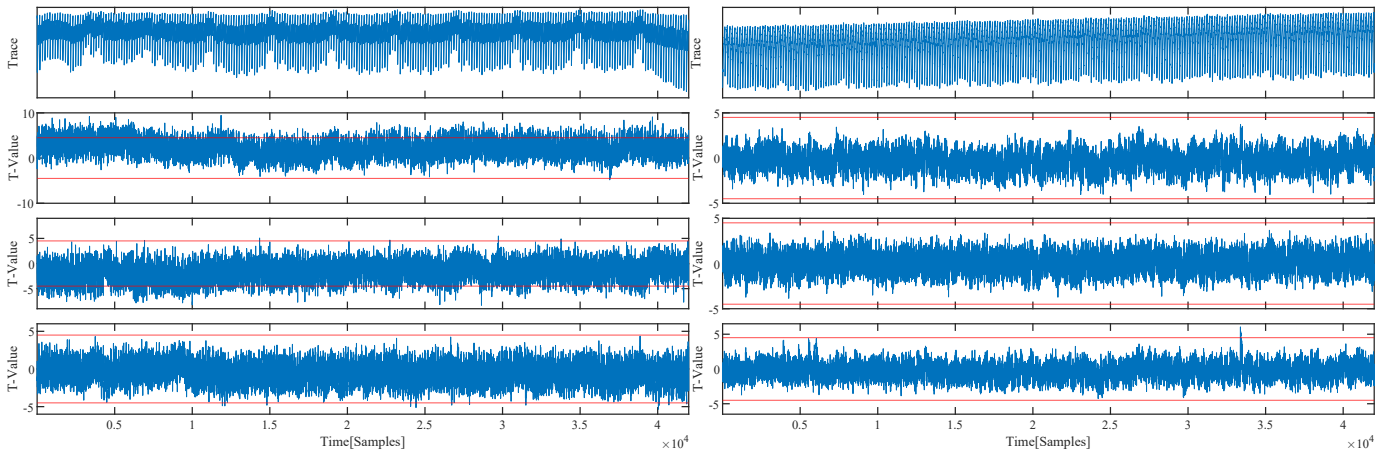

Fig. 7. Non-specific TVLA test on full duration of the second-order multiplicative masked M&M-AES. Left column: PRNG-off with 100,000 traces. Right column: PRNG-on with 50 million traces. Rows(top to bottom): average power trace, first-order, second-order, and third-order t-statistic.

It is worth mentioning that without modifying the fundamental structure of the M&M, our proposal inherits the vulnerability against IFA attacks. A recently reported SIFA2-like attack [20] on the M&M does not apply to our proposal due to the different S-Box architecture. However, the possibility of other similar attacks cannot be ruled out. We will further investigate the issue and perfect its security in the future.

REFERENCES

[1] L. De Meyer, V. Arribas, S. Nikova, V. Nikov, and V. Rijmen, "M&M: Masks and Macs against Physical Attacks," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 25–50, Nov. 2018.

[2] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Advances in Cryptology — CRYPTO '97*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, vol. 1294, pp. 513–525.

[3] S. Nikova, C. Rechberger, and V. Rijmen, "Threshold Implementations Against Side-Channel Attacks and Glitches," in *Information and Communications Security*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, vol. 4307, pp. 529–545.

[4] L. De Meyer, O. Reparaz, and B. Bilgin, "Multiplicative Masking for AES in Hardware," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 431–468, Aug. 2018.

[5] G. Goos, J. Hartmanis, J. Van Leeuwen, P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *Advances in Cryptology — CRYPTO' 99*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, vol. 1666, pp. 388–397.

[6] E. Brier, C. Clavier, and F. Olivier, "Correlation Power Analysis with a Leakage Model," in *Cryptographic Hardware and Embedded Systems - CHES 2004*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, vol. 3156, pp. 16–29.

[7] H. Gross, S. Mangard, and T. Korak, "Domain-Oriented Masking: Compact Masked Hardware Implementations with Arbitrary Protection Order," in *Proceedings of the 2016 ACM Workshop on Theory of Implementation Security*. Vienna Austria: ACM, Oct. 2016, pp. 3–3.

[8] Y. Li, K. Sakiyama, S. Gomisawa, T. Fukunaga, J. Takahashi, and K. Ohta, "Fault Sensitivity Analysis," in *Cryptographic Hardware and Embedded Systems, CHES 2010*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 320–334.

[9] C. Clavier, "Secret External Encodings Do Not Prevent Transient Fault Analysis," in *Cryptographic Hardware and Embedded Systems - CHES 2007*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, vol. 4727, pp. 181–194.

[10] T. Schneider, A. Moradi, and T. Güneysu, "ParTI – Towards Combined Hardware Countermeasures Against Side-Channel and Fault-Injection Attacks," in *Advances in Cryptology – CRYPTO 2016*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, vol. 9815, pp. 302–332.

[11] O. Reparaz, L. De Meyer, B. Bilgin, V. Arribas, S. Nikova, V. Nikov, and N. Smart, "CAPA: The Spirit of Beaver Against Physical Attacks," in *Advances in Cryptology – CRYPTO 2018*. Cham: Springer International Publishing, 2018, vol. 10991, pp. 121–151.

[12] D. Toprakhisar, S. Nikova, and V. Nikov, "CAPABARA: A Combined Attack on CAPA," in *Constructive Side-Channel Analysis and Secure Design*. Cham: Springer Nature Switzerland, 2024, vol. 14595, pp. 76–89.

[13] M.-L. Akkar and C. Giraud, "An Implementation of DES and AES, Secure against Some Attacks," in *Cryptographic Hardware and Embedded Systems — CHES 2001*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, vol. 2162, pp. 309–318.

[14] J. D. Golić and C. Tymen, "Multiplicative Masking and Power Analysis of AES," in *Cryptographic Hardware and Embedded Systems - CHES 2002*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, vol. 2523, pp. 198–212.

[15] L. Genelle, E. Prouff, and M. Quisquater, "Thwarting Higher-Order Side Channel Analysis with Additive and Multiplicative Maskings," in *Cryptographic Hardware and Embedded Systems – CHES 2011*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, vol. 6917, pp. 240–255.

[16] H. Gross, S. Mangard, and T. Korak, "An Efficient Side-Channel Protected AES Implementation with Arbitrary Protection Order," in *Topics in Cryptology – CT-RSA 2017*. Cham: Springer International Publishing, 2017, vol. 10159, pp. 95–112.

[17] J. Boyar, P. Matthews, and R. Peralta, "Logic Minimization Techniques with Applications to Cryptology," *Journal of Cryptology*, vol. 26, no. 2, pp. 280–312, Apr. 2013.

[18] G. Barthe, S. Belaïd, F. Dupressoir, P.-A. Fouque, B. Grégoire, P.-Y. Strub, and R. Zucchini, "Strong Non-Interference and Type-Directed Higher-Order Masking," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 116–129.

[19] T. Schneider and A. Moradi, "Leakage Assessment Methodology: A Clear Roadmap for Side-Channel Evaluations," in *Cryptographic Hardware and Embedded Systems – CHES 2015*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, vol. 9293, pp. 495–513.

[20] H. Hirata, D. Miyahara, V. Arribas, Y. Li, N. Miura, S. Nikova, and K. Sakiyama, "All You Need Is Fault: Zero-Value Attacks on AES and a New Lamda-Detection M&M," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2024, no. 1, pp. 133–156, Dec. 2023.

[21] S. Dhooghe and S. Nikova, "My Gadget Just Cares for Me - How NINA Can Prove Security Against Combined Attacks," in *Topics in Cryptology – CT-RSA 2020*, S. Jarecki, Ed. Springer International Publishing, 2020, vol. 12006, pp. 35–55.

[22] J. Feldtkeller, J. Richter-Brockmann, P. Sasdrich, and T. Güneysu, "CINI MINIS: Domain Isolation for Fault and Combined Security," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2022, pp. 1023–1036.