Reliable Inversion in GF(2^8) With Redundant Arithmetic for Secure Error Detection of Cryptographic Architectures

Mehran Mozaffari Kermani, *Senior Member, IEEE*, Amir Jalali, Reza Azarderakhsh, *Member, IEEE*, Jiafeng Xie, *Member, IEEE*, and Kim-Kwang Raymond Choo, *Senior Member, IEEE*

***Abstract*—In secure cryptographic primitives, such as block ciphers, the reliability of hardware implementations needs to be closely considered because faults in the hardware implementations can potentially reduce or impact on the underlying security. In this paper, we present approaches to detect errors in hardware implementations of the inversion in GF(2^8). The proposed approaches are based on both nonredundant and redundant arithmetic, utilizing normal basis (nonredundant) and two redundant Galois field representations, i.e., polynomial ring representation and redundantly represented basis through tower fields. To the best of our knowledge, this is the first work focusing on the error detection architectures for redundant arithmetic-based inversion in GF(2^8). The presented signature-based schemes in this paper are general and can be applied to block ciphers with 8-bit S-boxes, such as Camellia, SMS4, the advanced encryption standard, and CLEFIA. We present the results of error simulations and application-specific integrated circuit implementations to demonstrate the utility of the presented schemes. Based on the specific implementation's security/reliability objectives and the overhead/degradation tolerance for implementation/performance metrics, one can fine-tune and tailor the proposed work to achieve more reliable inversions in GF(2^8).**

***Index Terms*—Application-specific integrated circuit (ASIC), block cipher, error detection, inversion in GF(2^8), reliability.**

I. Introduction

CRYPTOGRAPHIC block ciphers are effective solutions for ensuring confidentiality in resource-constrained applications, e.g., deeply embedded systems in human body, such as sensitive implantable and wearable medical devices [1]. In lightweight cryptography, the S-boxes are usually small whose implementations can be achieved efficiently. The S-box sizes in lightweight block ciphers are usually with the size of 4-input, 4-output and can be implemented efficiently through logic gates or look-up tables. In applications, where memory macros on application-specific integrated circuit (ASIC) and block memories on field-programmable gate array (FPGA) are utilized, one can realize such look-up tables efficiently. Such variants of S-boxes typically have high performance but also require a larger area and a higher power consumption. On the other hand, there have also been extensive work on the realization of S-boxes based on multiplicative inversion in GF(2^8) used for cryptographic algorithms, such as the advanced encryption standard (AES) [2], Camellia [3], SMS4 [4], and CLEFIA [5].

S-boxes need to be implemented efficiently and can be realized through two different (and diverse) approaches in hardware. To alleviate such limitations, on the other hand, one can utilize composite field-based architectures. The latter have low complexity, but require subpipelining to achieve high throughput and efficiency. For high-performance applications, e.g., server or game console security, the implementations of the S-boxes can be done on FPGAs and through look-up table-based approaches to achieve high speed and performance. Nevertheless, such schemes can be replaced with composite field implementations to achieve low area and power consumption for deeply embedded systems. However, for either of these implementations, there might exist naturally occurring and malicious faults which undermine the reliability of block ciphers; consequently, impacting on their capability to assure confidentiality. A number of concurrent error detection techniques have been proposed to account for reliable hardware architectures (including cryptographic block ciphers) [6]–[19]. Such schemes are based on hardware/information/time/hybrid redundancy. Nevertheless, it is possible to leverage the error detection schemes which are tailored for specific implementations of the S-boxes to strike a balance between reliability and overhead tolerance.

The multiplicative inversion in GF(2^8) through composite fields can be realized using polynomial basis, normal basis, or mixed basis [20]–[22]. Although mixed basis can

Manuscript received January 1, 2017; revised March 23, 2017 and May 27, 2017; accepted June 9, 2017. Date of publication June 20, 2017; date of current version February 16, 2018. This work was supported by the National Institute of Standards and Technology through the U.S. Federal Agency under Award 60NANB16D245. This paper was recommended by Associate Editor Q. Xu. *(Corresponding author: Mehran Mozaffari Kermani.)*

M. Mozaffari Kermani is with the Department of Computer Science and Engineering, University of South Florida, Tampa, FL 33620 USA (e-mail: mmozaff@gmail.com).

A. Jalali and R. Azarderakhsh are with the Department of Computer and Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, FL 33431 USA (e-mail: ajalali2016@fau.edu; razarderakhsh@fau.edu).

J. Xie is with the Department of Electrical Engineering, Wright State University, Dayton, OH 45435 USA (e-mail: jiafeng.xie@wright.edu).

K.-K. R. Choo is with the Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249 USA (e-mail: raymond.choo@fulbrightmail.org).

Digital Object Identifier 10.1109/TCAD.2017.2717791

achieve high-performance implementations for the S-boxes, in [23], redundant basis is used for the S-boxes in composite fields to ameliorate the efficient realization of the architectures (40% higher efficiency in terms of area-time product compared to other bases). In this paper, we propose diagnosis approaches for the multiplicative inversion in GF(2^8). We note that permanent faults are caused by very-large-scale integration manufacturing defects (and of course if the intention is to break the entire device, such faults can be injected at run-time). On the other hand, "long transient faults" can lead to information leakage [24]. Simple time redundancy cannot detect long transient faults that last for the normal computation and recomputation, and it has been shown that one could successfully inject long transient faults to circumvent this countermeasure [24]. Our main contributions are summarized as follows.

1) For the first time, we propose error detection approaches for the multiplicative inversion in GF(2^8) utilizing combined normal basis (nonredundant) and two redundant Galois field representations, i.e., polynomial ring representation and redundantly represented basis through tower fields. One contribution of this paper is that the proposed approaches can be tailored based on the reliability requirements and overhead tolerance of sensitive usage models (not just confined to parities, for instance, or not getting confined to a certain number of blocks for deriving the signatures).
2) The focus of this paper is on presenting error detection architectures for multiplicative inversion, applicable to the S-boxes in a number of cryptographic algorithms, i.e., not only the AES [2] can benefit from the presented approaches, but Camellia [3], SMS4 [4], and CLEFIA [5] can also utilize them. We also present formulations for the transformation matrices to detect errors in their respective structures.
3) We benchmark the proposed architectures to evaluate their capability to detect transient and permanent faults using fault injection simulations. Findings demonstrate that the proposed architectures have acceptable error detection capabilities, i.e., reliability of proposed approaches.
4) Another contribution of this paper is to assess the false alarm resiliency of the proposed approaches. Using the proposed approaches, the error detection structures are capable of detecting the injected faults with high coverage (transient and permanent as well as single, multiple, and adjacent faults) with low false alarms.
5) We implement the proposed error detection architectures on an ASIC platform using a 65-nm standard-cell library. Our results show that the proposed efficient error detection architectures can be practically utilized, and is suitable for deployment on resource-constrained applications.

The organization of this paper is as follows. Section II presents background materials and our proposed error detection schemes. In Section III, we present concrete constructions. We then evaluate the performance of the proposed scheme followed by fault-injection simulations in Section IV. Finally, the conclusions are presented in Section V.

II. Proposed Error Detection Approaches

In what follows, we present the background materials and then the proposed error detection schemes.

A. Background Concepts

We will now briefly describe the background materials required in the understanding of the design of the multiplicative inversion in GF(2^8) based on tower field arithmetic.

Limitations of look-up table-based realization of the S-boxes include high power consumption, large area requirement, and high energy usage. On the other hand, composite field-based architectures have low complexity but pipelining is required to achieve high throughput and efficiency. The tower field approach for inversion in GF(2^8) is efficient because the subfields GF($((2^2)^2)^2$ and GF($(2^4)^2$ operations are realized efficiently. Polynomial/normal/mixed bases can be used for such tower fields (using $m = 8$ bits in a nonredundant manner).

For the nonzero element $a \in$ GF(2^8), we have the inverse $a^{-1} = a^{254}$. The scheme underlying the tower field approach is to utilize smaller arithmetic operations over subfield GF($((2^2)^2)^2$ or GF($(2^4)^2$ instead of GF(2^8). There is an isomorphism between the elements of GF(2^8) and those of the tower field. Such a multiplicative inversion in GF(2^8) can be efficiently implemented using the Itoh–Tsujii algorithm [25]. For instance, for the subfield GF($((2^2)^2)^2$ with normal basis composite field, let a field element a in GF($((2^2)^2)^2$ be the input given by $h\alpha^{16} + l\alpha$ in normal basis with h and l being the upper and lower 4-bit parts of a, respectively. We derive the inversion of a by the calculation of the 16th and 17th powers, subfield inversion, and the final multiplication.

In this paper, we use redundant-based representation, which uses $n(> m = 8)$ bits to represent each element of GF(2^8). The intention here is to show that we derive efficient schemes for error detection in such tower fields. We calculate the inversion of the tower field GF($(2^4)^2$ by the normal basis because the squaring operation is performed solely by wiring and two redundant-based representations combination [23].

B. Error Detection for Redundantly Represented Basis

Each element in such basis is represented by a root of an mth degree irreducible polynomial. Similar to normal basis, redundantly represented basis-based GF(2^8) squaring can be performed by bit-wise permutation. We note that such basis can be obtained by adding a base, β^0, to optimal normal basis. For squaring, we propose using signatures, e.g., parities or interleaved parities, as they remain intact for the actual and predicted variants, leading to very efficient error detection.

We present Theorems 1 and 2 for the error detection of the multiplication in GF(2^4) in redundantly represented basis.

Theorem 1: The predicted parity for the multiplication in GF(2^4) in redundantly represented basis is as follows (s and t are 5-bit entries and the result u is also 5 bits;

$s = s_4\beta^4 + s_3\beta^3 + s_2\beta^2 + s_1\beta + s_0$, $t = t_4\beta^4 + t_3\beta^3 + t_2\beta^2 + t_1\beta + t_0$, $u = u_4\beta^4 + u_3\beta^3 + u_2\beta^2 + u_1\beta + u_0$)

$$\hat{P} = \sum_{0 \le i,j \le 4, i \ne j} s_i.t_j \tag{1}$$

where the "sum" $\sum$ symbol represents XOR operation, β is the root of the fourth degree all-one polynomial, and $s_i, t_j \in$ GF(2).

Proof: We have the following for the result of the multiplication in GF(2^4) in redundantly represented basis:

$$u_0 = (s_1 + s_4)(t_1 + t_4) + (s_2 + s_3)(t_2 + t_3) \tag{2}$$
$$u_1 = (s_0 + s_1)(t_0 + t_1) + (s_2 + s_4)(t_2 + t_4) \tag{3}$$
$$u_2 = (s_0 + s_2)(t_0 + t_2) + (s_3 + s_4)(t_3 + t_4) \tag{4}$$
$$u_3 = (s_0 + s_3)(t_0 + t_3) + (s_1 + s_2)(t_1 + t_2) \tag{5}$$
$$u_4 = (s_0 + s_4)(t_0 + t_4) + (s_1 + s_3)(t_1 + t_3). \tag{6}$$

Modulo-2 addition of the above results in the predicted parity of $\hat{P} = (s_1+s_4)(t_1+t_4)+(s_2+s_3)(t_2+t_3) +(s_0+s_1)(t_0+t_1)+(s_2+s_4)(t_2+t_4)+(s_0 +s_2)(t_0+t_2)+(s_3+s_4)(t_3+t_4)+(s_0+s_3)(t_0+t_3)+(s_1+s_2)(t_1+t_2)+(s_0+s_4)(t_0 +t_4)+(s_1+s_3)(t_1+t_3) = s_0(t_1 + t_2 + t_3 +t_4) + s_1(t_0 + t_2 + t_3 + t_4) + s_2(t_1 + t_0 + t_3 +t_4) + s_3(t_1 + t_2 + t_0 + t_4) + s_4(t_1 + t_2 + t_3 + t_0)$ which is shown for the sake of simplicity as (1). ■

Remark 1: The predicted parity $\hat{P} = s_0(t_1+t_2+t_3+t_4)+s_1(t_0 +t_2 + t_3 + t_4) + s_2(t_1 + t_0 + t_3 + t_4) + s_3(t_1 + t_2 + t_0 + t_4) + s_4(t_1 + t_2 + t_3 + t_0)$ can be implemented using 13 XOR gates and 5 AND gates with the critical path delay of 5 T_X and 1 T_A, where T_X and T_A are the delays for XOR and AND gates, respectively.

Theorem 2 describes the interleaved parities for burst error detection.

Theorem 2: The predicted interleaved parities for the multiplication in GF(2^4) in redundantly represented basis are as follows ($\hat{P}_1$ and $\hat{P}_2$ represent, respectively, $u_0 + u_2 + u_4$ and $u_1 + u_3$) : $\hat{P}_1 = s_0(t_2 + t_4) + s_1(t_3 + t_4) +s_2(t_0 + t_3) + s_3(t_1 + t_2 + t_3 + t_4) + s_4\ (t_0 + t_1 + t_3 + t_4), \hat{P}_2 = s_0(t_1 + t_3) + s_1(t_0 + t_2) + s_2(t_1 + t_4) + s_3(t_0 + t_3) + s_4(t_2 + t_4)$.

Proof: Considering the proof of Theorem 1, we derive the interleaved parities for $\hat{P}_1$ and $\hat{P}_2$, respectively, which can be efficiently implemented through subexpression sharing. ■

Remark 2: The predicted interleaved parities $\hat{P}_1 = s_0(t_2 + t_4) + s_1(t_3 + t_4) + s_2(t_0 + t_3) + s_3(t_1 + t_2 + t_3 + t_4) + s_4(t_0 + t_1 + t_3 + t_4)$, $\hat{P}_2 = s_0(t_1 + t_3) + s_1(t_0 + t_2) + s_2(t_1 + t_4) + s_3(t_0 + t_3) + s_4(t_2 + t_4)$ can be implemented for the former using 10 XOR gates and 5 AND gates with the critical path delay of 5 T_X and 1 T_A, and for the latter using 9 XOR gates and 5 AND gates with the critical path delay of 4 T_X and 1 T_A. One can save 2 XOR gates with further sharing the architectures of the two interleaved signatures.

C. Error Detection for Subfield Inversion in GF(2^4)

The inversion in GF(2^4) is constructed using the inputs in polynomial ring representation and the outputs in redundantly represented basis. We now present fault diagnosis schemes for the inversion unit noting that the transformation matrix used for basis conversion is implemented without incurring additional cost.

We present Theorem 3 for fault diagnosis of the 5-bit input ($d = d_4\beta^4 + d_3\beta^3 + d_2\beta^2 + d_1\beta + d_0$) and its inversion, i.e., $e = e_4\beta^4 + e_3\beta^3 + e_2\beta^2 + e_1\beta + e_0$.

Theorem 3: The predicted parity and the interleaved parities for the inversion in GF(2^4) are derived as follows:

$$\hat{P} = d_1d_2d_3\overline{(d_0 + d_4)} + \overline{d_0}(d_2d_3d_4 + d_1d_3d_4 + d_1d_2d_4) \tag{7}$$

$$\hat{P}_1 = d_0d_3d_4\overline{d_2} + d_2d_1d_0\overline{d_4} + (d_2 + d_3)\overline{d_1} + (d_2 + d_1)d_4 \tag{8}$$

$$\hat{P}_2 = \overline{d_0}d_1d_2\overline{d_3} + d_2d_3d_4\overline{d_1} + d_3\overline{d_1}d_4\overline{d_0} + d_1d_4\overline{d_2} + d_2 \vee d_3. \tag{9}$$

Proof: We have the following for the result of the inversion in GF(2^4):

$$e_0 = (d_1 \vee d_4)(d_2 \vee d_3) \tag{10}$$
$$e_1 = \overline{d}_4(d_1 + d_2) \vee (d_0d_4(d_2 \vee d_3)) \tag{11}$$
$$e_2 = \overline{d}_3(d_2 + d_4) \vee (d_0d_3(d_1 \vee d_4)) \tag{12}$$
$$e_3 = \overline{d}_2(d_1 + d_3) \vee (d_0d_2(d_1 \vee d_4)) \tag{13}$$
$$e_4 = \overline{d}_1(d_3 + d_4) \vee (d_0d_1(d_2 \vee d_3)). \tag{14}$$

Modulo-2 addition of the above formulas as well as its interleaved modulo-2 addition results in (7)–(9) after simplifications. ■

Remark 3: The predicted (interleaved) parities $\hat{P} = d_1d_2d_3\overline{(d_0 + d_4)} + \overline{d_0}(d_2d_3d_4 + d_1d_3d_4 + d_1d_2d_4)$, $\hat{P}_1 = d_0d_3d_4\overline{d_2} + d_2d_1d_0\overline{d_4} + (d_2 + d_3)\overline{d_1} + (d_2 + d_1)d_4$, $\hat{P}_2 = \overline{d_0}d_1d_2\overline{d_3} + d_2d_3d_4\overline{d_1} + d_3\overline{d_1}d_4\overline{d_0} + d_1d_4\overline{d_2} + d_2 \vee d_3$ can be implemented, respectively, excluding the inverters, using 3 XOR gates and 9 AND gates with the critical path delay of 3 T_X and 3 T_A, using 5 XOR gates and 8 AND gates with the critical path delay of 2 T_X and 2 T_A, and finally utilizing 4 XOR gates and 12 AND/OR gates with the critical path delay of 3 T_X and 2 T_A.

D. Error Detection for Other Stages

The other stages construct the beginning of the architecture (assuming h and l as the 4-bit entries to the entire inversion in GF($(2^4)^2$) and d as the 5-bit input of the inversion in GF(2^4)). Considering the mappings from normal basis and polynomial ring representation, we present Theorem 4 for the parity prediction of the stage which performs $\phi' hl + \phi''(h+l)^2$, where we have

$$\phi' \rightarrow \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix} \text{ and } \phi'' \rightarrow \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix}.$$

Theorem 4: The predicted parity for $\phi' hl + \phi''(h + l)^2$ is efficiently derived with no cost as $\hat{P} = 0$. This yields to very lightweight architectures using such derivation and thus using interleaved parities is not recommended for this stage unless a specific error coverage is aimed.

Proof: Due to page limitations, we do not present the formulations for the result of $\phi' hl + \phi''(h + l)^2$, whose modulo-2 addition of the output bits results in $\hat{P} = 0$. ■

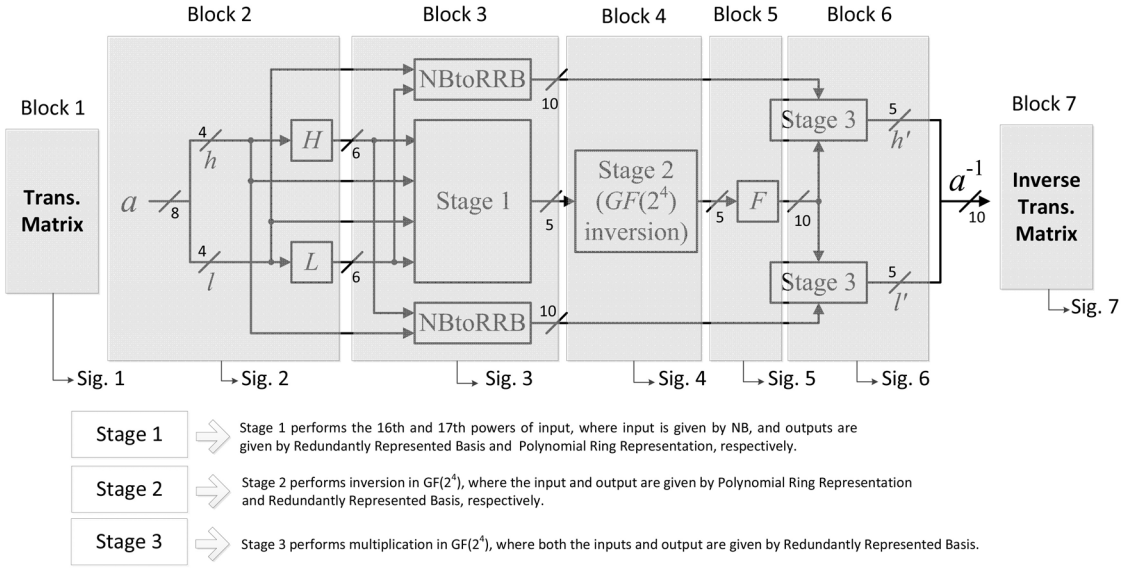

Fig. 1. First proposed choice in error detection of the inversion in GF($(2^4)^2$).

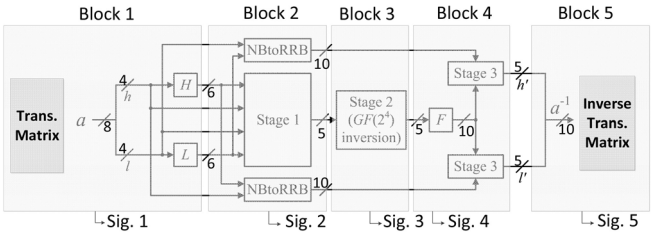

Fig. 2. Second proposed choice in error detection of the inversion in GF($(2^4)^2$).

To finalize this section, we would like to emphasize that the utilized signatures in this paper are a subset of possible ones that can be used. For instance, one may use a three-flag, three-bit signature for different subparts of the S-boxes in redundant-basis, such as the inversion in GF(2^4) as $S_0 = e_0 + e_1$, $S_1 = e_2 + e_3$, and $S_1 = e_4$. Such derivation, and other similar ones, would add to the overhead of the error detection architecture, with the advantage of higher error coverage; nevertheless, based on the reliability requirements and overhead tolerance, such schemes can also be tailored for error detection (this also depends on the fault model used).

III. Efficient Detection Construction

In this section, using the derived formulations for the signatures in this paper, we present the error detection constructions and based on four different considerations, namely; complexity (area/delay and, thus, the respective power/energy), error coverage, false alarms, and real attacker resiliency. We then choose an efficient construction from the pool of designs in Figs. 1–3. Error simulations and ASIC implementations are presented for these constructions in the next section.

A. Three Choices to Consider

Based on the derived signatures, we examine three choices for fault diagnosis as shown in Figs. 1–3. The proposed schemes are not confined to specific number of blocks (although three of such choices are shown in Figs. 1–3). We also remark that for different reliability requirements and overhead tolerance, one can fine-tune such choices. There are two reasons for choosing these three constructions as examples for presenting the proposed error detection schemes. First, the choices are made to preserve the inner operations in composite field and having them intact, allowing them to be implemented through different inner-structures. Second, noting that different usage models and contexts constrain the error detection architectures in terms of the required error coverage and overhead tolerance, we have made sure that these examples cover three cases to have such respective compromise. In addition to these three examples, we have also simulated for assessing the error coverage and implemented on ASIC another case study (with four blocks by modifying Block 1 in Fig. 3 to two blocks, one being the transformation matrix). As it is shown in our simulations and through our implementations, the error coverages of Figs. 1 and 2 and the four-block construction are higher, in the order specified, compared to the construction in Fig. 3. The false alarms, although negligible, decrease in the same order. However, the overhead and performance degradation of the construction in Fig. 3 are the lowest. Finally, we note that having many blocks increases the error detection overhead unreasonably and, thus, might not be preferred. On the other hand, having only one block would result in low error coverage, and, in addition, it is interesting that the resulting formulations for signatures would be costly causing high overhead which is not preferred. Thus, the eventual choice is dependent on the reliability requirements and overhead tolerance of the architectures.

As observed in Fig. 1, in the first choice, seven sub-blocks (blocks 1–7) are used for error detection whose predicted signatures are based on the formulations in the preceding section. Let us choose the parity for finding an error detection construction, noting that this does not confine us to this signature only.

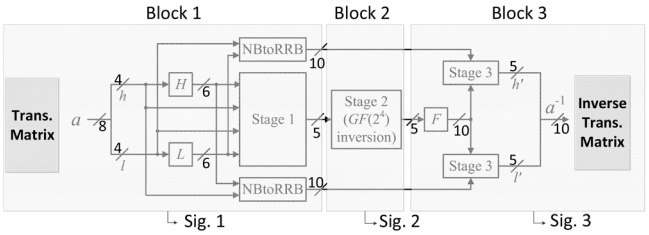

Fig. 3. Third proposed choice in error detection of the inversion in GF($(2^4)^2$).

In Fig. 1, the predicted signatures (parities for example) are shown as Sig. 1–7.

The derivation of the multiplicative inversion is performed as follows.

1) The 8-bit input a is divided into lower and higher halves (4-bit l and h, respectively) and as entries to H and L units (which derive the 6-bit outputs by having $H_{i,j} = h_i + h_j, L_{i,j} = l_i + l_j (1 \leq i < j \leq 4)$). Similar definition is used for the unit F. Stage 1 performs the 16th and 17th powers of input, where input a is given by normal basis, and outputs a^{16} and a^{17} are given by redundantly represented basis and polynomial ring representation, respectively. For α as the root of a second degree irreducible polynomial over GF(2^4), with respective irreducible polynomials, we get the 16th and 17th powers of input ($a^{16} = l\alpha^{16} + h\alpha, a^{17} = hl\mu^2 + (h+l)^2\upsilon$). Stage 1 performs $\phi' hl + \phi''(h+l)^2$, where the linear matrices used, ϕ' and ϕ'', are explained in Theorem 4.
2) The transformation matrices, as well as the "free" normal basis to redundant basis conversion (because the normal basis can be considered as the reduced version of redundantly represented basis with the same root of the fourth degree all-one polynomial), are also shown in Fig. 1. Stage 2 performs the subfield inversion (inputs and outputs are in redundant basis), where the input and output are given by polynomial ring representation and redundantly represented basis, respectively. More details are presented in Theorem 3.
3) In Stage 3 (two subparts as seen in Fig. 1), two multiplications in redundant basis (redundantly represented basis) are computed. More details are presented in Theorem 1. Figs. 2 and 3 show the same structure but with different number of check points, i.e., 5 and 3, respectively.

A single gate, e.g., the XOR gate used for comparing the predicted/actual signatures, can be separately hardened using logical or circuit-level fault tolerance techniques. Finally, we note that one can use a combination of the proposed methods to achieve the intended error coverage, alleviate the complications, such as detecting the faults that occur right at the output of blocks, and fine-tuning the false alarm ratios.

B. False Alarms

False-alarms could impact on the utilization of cryptographic solutions. Specifically, if such alarms become repetitive, they might hinder the normal operations of cryptographic algorithms by inducing distrust in users who may eventually abandon the particular security solutions. Constructions in Figs. 1–3 have been simulated for errors (specifically for false alarms). Although the number of such alarms is not high, different constructions have different (and somewhat comparable) characteristics. For example, our error simulations show that Fig. 1 has the highest number of such alarms ($\simeq$0.05%) and Fig. 3 has the lowest number ($\simeq$0.03%). Because we are dealing with multiple signatures, such alarms are due to the cases in which we detect faults in an inner sub-block which will not be eventually translated into errors in the output.

C. Linear Transformations Within Ciphers

In this paper, we have focused on the 8-bit S-boxes which are the only nonlinear transformations within the ciphers and consume much of the area and constitute high power consumption of the ciphers. Moreover, a major motivation has been that the key schedule unit constitutes the S-boxes as the main transformation. Nevertheless, we briefly present here the error detection schemes of the linear transformations within ciphers as well.

Focusing on the AES, three transformations other than the S-boxes (also known as SubBytes) are: 1) ShiftRows; 2) MixColumns; and 3) AddRoundKey. The first one is just rewiring in hardware, and the derivation of any signature is straightforward. The second one is a linear transformation (multiplication with a constant matrix in finite field), for which, based on the error detection requirements and overhead tolerance, column signatures can be derived. Finally, AddRoundKey is implemented through XOR gates; thus, signature derivation is performed accordingly.

IV. Error Simulations and ASIC Synthesis

As seen in Figs. 1–3, one needs to derive the signatures for the transformation and inverse transformation matrices (corresponding to Sig. 1 and the signature of last block in these figures, respectively). We combine the last block with the affine transformation matrix of the AES to derive two matrices [23] to apply error detection according to Theorem 5.

Theorem 5: The predicted parity and interleaved parity for Block 1 ($\hat{P_{b1}}$) and "combined" last block (lb) with affine transformation ($\hat{P_{lb}}$) are derived as below, considering S as the input to Block 1, A as the output of Block 1, A^{-1} as the input to the last block, and O as the output of last block: $\hat{P_{b1}} = s_1 + s_2 + s_3 + s_4 + s_5 + s_6, \hat{P_{1b1}} = s_1, \hat{P_{2b1}} = s_2 + s_3 + s_4 + s_5 + s_6, \hat{P_{lb}} = a_1^{-1} + a_2^{-1}, \hat{P_{1lb}} = a_0^{-1} + a_1^{-1} + a_2^{-1} + a_3^{-1} + a_6^{-1} + a_7^{-1}, \hat{P_{2lb}} = a_0^{-1} + a_3^{-1} + a_6^{-1} + a_7^{-1}$.

Proof: The transformation matrix (M_1 below) is an 8×8 one, and the merged inverse and affine transformation (M_2 below) is an 8×10 matrix (see Figs. 1–3), through which the above is derived, noting that the least significant bit is on top and also the input to Block 1 is in normal basis

$$M_1 \rightarrow \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

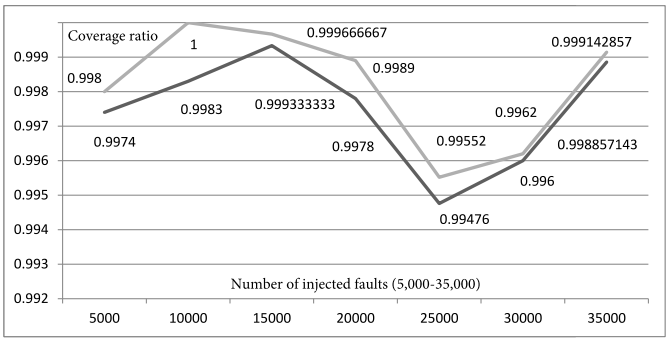

Fig. 4. Error detection ratios for interleaved parity (gray) and parity (black) for the proposed construction in Fig. 1 for up to 35 000 fault injections.

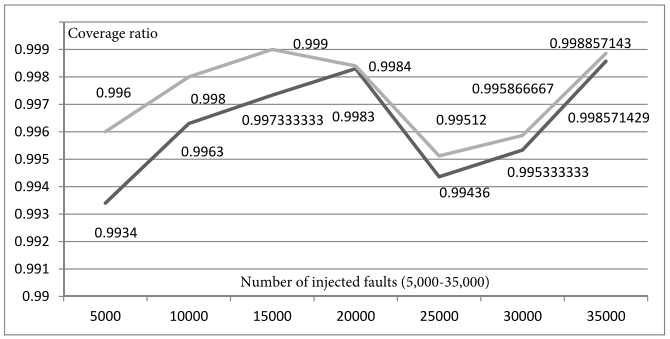

Fig. 5. Error detection ratios for interleaved parity (gray) and parity (black) for the proposed construction in Fig. 2 for up to 35 000 fault injections.

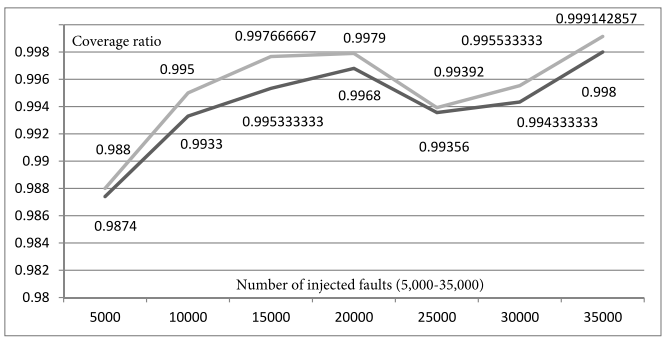

Fig. 6. Error detection ratios for interleaved parity (gray) and parity (black) for the proposed construction in Fig. 3 for up to 35 000 fault injections.

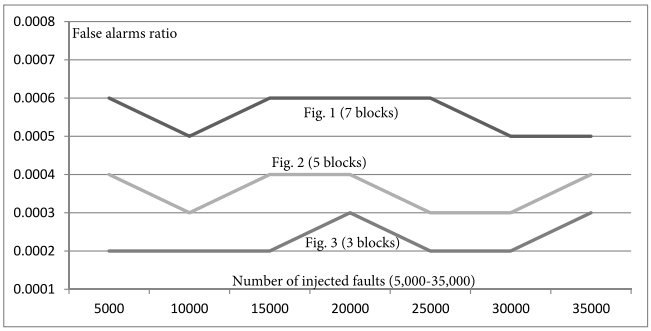

Fig. 7. False alarms ratios for the proposed three constructions for up to 35 000 fault injections.

$$M_2 \rightarrow \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}. \quad \blacksquare$$

The proposed error detection architectures (applied to the AES) have been simulated after injecting faults. The proposed architectures have the capability of detecting both permanent and transient faults (this covers both natural and malicious faults). The approach in the proposed fault diagnosis schemes is designed to inject faults and then observe the error indication flags. For simulations, Verilog HDL is used. We consider all sub-blocks of the original architecture to induce faults by flipping one or more bits and then inspect the generated outputs.

We consider a particular fault scenario and apply different inputs to assert a subset of entries while injecting faults. We then observe all detected errors for the inputs. The fault model used to test the proposed architectures is created using external feedback linear feedback shift registers to generate pseudo-random fault vectors that can flip random bits in the output of the gates and at random intervals. For the three architectures presented in Figs. 1–3, we inject 35 000 faults and record the number of errors as seen in Figs. 4–6. Moreover, in Fig. 7, the false alarm ratios are presented. As seen in Figs. 4–6, the error coverage in all the cases is more than 99% (for Fig. 1, we slightly have higher coverage as seen in Fig. 4; nevertheless, because we consider all the S-boxes in the SubBytes transformation, the detection ratios of the architectures presented are close). We note that in these figures, error coverage ratios are shown which is the number of detected faults over the total injections. The difference between the error detection results is, comparably, not high. In addition, the results of our simulations for 80 000 injected faults with the ratios of detected faults obtained as 0.9998, 0.9989, and 0.9991 for Figs. 1–3, respectively, have been obtained for parity as signature. These show that the slight fluctuations in Figs. 4–6 do not follow a trend. Finally, the false alarm ratios are typically low (see Fig. 7); however, if that is the usage model concern, then the architecture in Fig. 3 is preferable.

A. Proposed Approaches in Presence of Fault Attacks

We note that the fundamental difference between security attacks and random faults is the intelligent-attacker assumption. Injection of random faults mimics errors happening due to natural causes. In contrast, the intelligent adversary running fault-based cryptanalysis will carefully determine the fault (s)he is going to inject and perform injection right at the calculated position and point of time. Consequently, we note that just trying random faults will not be helpful in breaking most ciphers. As such, we emphasize that the signatures we use for error simulations and implementations are general and, for instance, for the case of parities, we detect all the odd errors (and single errors). Ideally, single stuck-at faults

are better for real attackers to obtain side-channel information. In reality, however, this might not happen due to lack of technological advancements. Differential fault analysis uses transient and mostly multiple bit and byte faults. In the case of interleaved parities, we detect burst faults which are more realistic to consider for the attackers. We note that a variant of fault attacks, i.e., fault sensitivity analysis, can be thwarted, for instance, through eliminating fault sensitivity based on a delay insertion algorithm.

B. Differential Fault Intensity Analysis

A subset of fault attacks, differential fault intensity analysis; see [26]–[28], combines differential power analysis with fault injection principles to obtain biased fault models. The advantage in such an approach is that the same fault in both the original and redundant computations can be injected, but not all faults occur with equal probability. In practice, the attacker is interested in using as few faults as possible (preferably single faults with different intensities) to minimize the effort. Previous work argue that the single-bit (more likely in low fault intensity), two-bit, three-bit, and four-bit (more likely in higher intensities) biased fault models can be used to simulate variation of fault intensity. In addition, fault categories presented in [29] include: single bit upset (SBU), single byte double bit upset, single byte triple bit upset (SBTBU), single byte quadruple bit upset (SBQBU), other single byte faults (OSB), and multiple byte faults (MB); the former four corresponding to single/two/three/four-bit models.

C. Flexibility in Choosing the Signatures

Compared to the approaches based on using 1-bit parity for the entire 16 S-boxes or those using 1-bit parities for one S-box (information redundancy), the proposed work gives flexibility to the designers in order to choose three instances of dividing the architectures into blocks (Figs. 1–3) and also have the flexibility of using single of interleaved parities. The proposed approaches in this paper based on error detecting codes are able to thwart a number of such fault models. Specifically, SBUs and SBTBUs are detected fully through the approaches based on error detecting codes and using parities. Moreover, through interleaved parities, in addition to burst faults, a number of SBTBUs, SBQBUs, OSBs, and MBs are detected. Error detecting codes and column signatures (parities) can also detect OSBs and MBs, based on our simulations. The proposed methods, being for reliability, can deal with permanent and transient faults and compound the challenge of launching fault analysis attacks. The signature-based diagnosis approach, which uses linear codes that can (always) detect random errors of small multiplicity (and can never detect some other errors), differs from an architecture based on robust codes which can detect (with probability) any error. We also note that plain time/hardware redundancy has the disadvantage of high overhead; however, the potential remedy for time redundancy, i.e., recomputing with encoded operands (hybrid redundancy), while being expensive, is able to detect permanent and transient faults similar to the proposed approach here.

D. Making the Attacks Difficult

We note that having the flexibility in choosing the signatures for the three architectures in Figs. 1–3 is an advantage for our schemes. It has been shown that it is practical to attack parity-based schemes that use merely 1-bit parity for SubBytes or even one S-box; however, the number of injections needed would be high if composite fields are used (which allow incorporating the proposed work). Multibyte faults cannot be used to realistically attack time redundancy countermeasure implementations, e.g., recomputing with permuted operands, and single-byte fault models are the only viable option for the attackers [29]. We remark that, however, unlike our proposed schemes, recomputing with permuted operands could fail to detect the occurrence of a fault as long as the adversary could inject the same fault in both the original and redundant computations (biased fault model makes it easier). Such recomputing with permuted operands can be used in conjunction with coding schemes to nullify the effect of the bias in the fault model by fault space transformation (if two equivalent faults f_0 and f_1 are injected in the output registers, we use a mapping that transforms the fault space); thus thwarting both these attack schemes. This is similar to the schemes used in [29], which incurs additional overhead.

E. ASIC Implementations and Comparisons

Through ASIC and for the constructions in Figs. 1–3 and the previous work using look-up table (LUT)-based and composite field-based S-boxes, we also present the performance and implementation metrics for the multiplicative inversion in GF(2^8) utilizing different structures as well as normal basis (nonredundant) and two redundant Galois field representations. The benchmarking is performed for the error detection architectures using TSMC 65 nm library and Synopsys Design Compiler (shown in Table I for area, frequency, power consumption at the respective working frequencies, and efficiency [throughput over gate equivalents (GE)]). There are two notes to consider regarding the synthesis.

1) We synthesized the multiplicative inversion in GF(2^8) which can be used for cryptographic algorithms, such as the AES [2], Camellia [3], SMS4 [4], CLEFIA [5], and the like.
2) We mainly compare signature-based approaches; nevertheless, approaches based on recomputations, e.g., NREPO (normal basis recomputing with encoded operands which has been proven to be resilient against fault analysis attacks at the cost of higher overhead), can be combined with the proposed architectures.

In Table I, the first two schemes are based on LUTs and are expensive on ASIC and the overhead difference compared to the other schemes is high. Moreover, the scheme based on double-data-rate using both clock edges of registers is presented. This is achieved at the expense of suffering from high throughput degradation. We note that in Table I, in order to make the area results meaningful when switching technologies, we have also provided the NAND-gate equivalency (GE). This is performed using the area of a NAND gate in the utilized TSMC 65-nm CMOS library which is 1.41 μm^2. As reported in the table, our proposed work outperforms other

TABLE I
OVERHEADS AND PERFORMANCE DEGRADATION COMPARISON FOR THE MULTIPLICATIVE INVERSION IN GF(2^8) OF THE S-BOXES INCLUDING 7/5/3-BLOCK ERROR DETECTION (ED) IN FIGS. 1–3 ON 65-nm ASIC

Error detection structure	Area μm^2 (GE)	Diff.	Freq. $[MHz]$	Diff.	Power (μW)	Diff.	Effic.[6]	Diff.
Redundancy/LUT (+affine) [17][1]	$\simeq$36k (26k)	N/A	642	N/A	7,432	N/A	0.19	N/A
Parity, 256×9 LUT (+affine) [19][1]	$\simeq$19k (14k)	N/A	1,198	N/A	3,012	N/A	0.7	N/A
Double-Data-Rate [9][2]	606 (430)	86.9%	298	47.6%	169	128%	5.4	71.8%
Robust codes [13][3]	518 (368)	60%	289	41.9%	118	59.4%	6.3	67.2%
NREPO [18][4]	502 (356)	54.7%	260	52.9%	105	41.9%	5.5	71.3%
Polynomial basis [20] with 5-block ED	513 (364)	58.2%	303	45.1%	113	52.7%	6.6	65.6%
Normal basis [21] with 5-block ED	430 (305)	32.6%	312	43.5%	98	32.4%	8.2	57.3%
Mixed basis [22] with 5-block ED	497 (353)	53.4%	476	13.7%	121	63.5%	10.8	43.7%
Redundant basis (this work/7-block ED)	439 (312)	35.6%	495	10.3%	88	18.9%	12.7	33.8%
Redundant basis (this work/5-block ED)	402 (285)	23.9%	498	9.7%	82	10.8%	14.0	27%
Redundant basis (this work/3-block ED)	406 (288)	25.2%	498	9.7%	84	13.5%	13.9	27.6%
Original redundant basis[5]	324 (230)	B/line	552	B/line	74	B/line	19.2	B/line

1. These two schemes are based on LUTs and thus are expensive on ASIC and the overhead difference compared to the other schemes is high, noted by N/A.
2. This is based on double-data-rate using both clock edges of registers. All registers are duplicated and extra MUXes are added.
3. Robust codes (compared to other signatures) detect with high probability (not usually 100%) a number of fault models.
4. Normal basis recomputing with encoded operands (proven to be effective against fault analysis attacks at the expense of suffering from high throughput degradation).
5. This is used as the baseline and all the overheads and degradations are with respect to this scheme.
6. Efficiency ($Mbps/GE$) is calculated by dividing "throughput" by "area". Some benchmarks use inverse of "time-area" instead of this measure which is directly correlated.

approaches, in terms of performance and implementation metrics, making it suitable for efficient error detection of the multiplicative inversion in GF(2^8). For the entire AES encryption (which includes 16 S-boxes in each of its ten rounds), the ASIC synthesis GE area results for redundant basis (this paper/Figs. 1–3) are 61 243 GE, 57 300 GE, and 56 008 GE, respectively. One future research direction is to investigate combined fault and power analysis attacks countermeasures for the multiplicative inversion in GF(2^8) using redundant basis (related prior work, for instance, [14], have not considered such basis).

V. CONCLUSION

In this paper, signature-based error detection approaches were presented for the multiplicative inversion in GF(2^8) utilizing normal basis (nonredundant) and two redundant Galois field representations, i.e., polynomial ring representation and redundantly represented basis through tower fields. In our approaches, we considered both the reliability and performance metrics objectives. Signature-based approaches are used for such nonlinear blocks to achieve high efficiency, while maintaining high error coverage. Our results demonstrated that the proposed efficient error detection architectures can be feasibly utilized, suitable for the required performance, reliability, and implementation metrics for constrained applications. As observed in Figs. 4–6, the error coverage in all the cases is more than 99%. Moreover, as reported in Table I, better performance and implementation metrics were achieved in the proposed work. This makes it suitable for efficient error detection of the multiplicative inversion in GF(2^8).

REFERENCES

[1] M. Mozaffari Kermani, M. Zhang, A. Raghunathan, and N. K. Jha, "Emerging frontiers in embedded security," in *Proc. Conf. VLSI Design*, Pune, India, Jan. 2013, pp. 203–208.

[2] *Specification of the Advanced Encryption Standard*. Accessed on Jul. 2017. [Online]. Available: http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[3] *Specification of Camellia, a 128-bit Block Cipher*. Accessed on Jul. 2017. [Online]. Available: https://info.isl.ntt.co.jp/crypt/eng/camellia/dl/01espec.pdf

[4] *Specification of SMS4*. Accessed on Jul. 2017. [Online]. Available: http://eprint.iacr.org/2008/329.pdf

[5] *Specification of CLEFIA*. Accessed on Jul. 2017. [Online]. Available: http://www.sony.net/Products/cryptography/clefia/index.html

[6] C. H. Yen and B. F. Wu, "Simple error detection methods for hardware implementation of Advanced Encryption Standard," *IEEE Trans. Comput.*, vol. 55, no. 6, pp. 720–731, Jun. 2006.

[7] G. Di Natale, M. Doulcier, M. L. Flottes, and B. Rouzeyre, "A reliable architecture for parallel implementations of the Advanced Encryption Standard," *J. Electron. Test. Theory Appl.*, vol. 25, no. 4, pp. 269–278, Aug. 2009.

[8] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "Concurrent structure-independent fault detection schemes for the Advanced Encryption Standard," *IEEE Trans. Comput.*, vol. 59, no. 5, pp. 608–622, May 2010.

[9] P. Maistri and R. Leveugle, "Double-data-rate computation as a countermeasure against fault analysis," *IEEE Trans. Comput.*, vol. 57, no. 11, pp. 1528–1539, Nov. 2008.

[10] X. Guo, D. Mukhopadhyay, C. Jin, and R. Karri, "Security analysis of concurrent error detection against differential fault analysis," *J. Cryptograph. Eng.*, vol. 5, no. 3, pp. 153–169, 2015.

[11] M. Yasin, B. Mazumdar, S. S. Ali, and O. Sinanoglu, "Security analysis of logic encryption against the most effective side-channel attack: DPA," in *Proc. Defect Fault Tolerance VLSI Syst.*, Amherst, MA, USA, 2015, pp. 97–102.

[12] M. Mozaffari-Kermani, R. Azarderakhsh, and A. Aghaie, "Reliable and error detection architectures of Pomaranch for false-alarm-sensitive cryptographic applications," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 12, pp. 2804–2812, Dec. 2015.

[13] M. Karpovsky, K. J. Kulikowski, and A. Taubin, "Robust protection against fault-injection attacks on smart cards implementing the Advanced Encryption Standard," in *Proc. Depend. Syst. Netw.*, 2004, pp. 93–101.

[14] H. Pahlevanzadeh, J. Dofe, and Q. Yu, "Assessing CPA resistance of AES with different fault tolerance mechanisms," in *Proc. Asia South Pac. Design Autom. Conf. (ASP DAC)*, 2016, pp. 661–666.

[15] T. Schneider, A. Moradi, and T. Güneysu, "ParTI—Towards combined hardware countermeasures against side-channel and fault-injection attacks," in *Proc. CRYPTO*, 2016, pp. 302–332.

[16] X. Guo and R. Karri, "Recomputing with permuted operands: A concurrent error detection approach," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 32, no. 10, pp. 1595–1608, Oct. 2013.

[17] R. Karri, K. Wu, P. Mishra, and K. Yongkook, "Fault-based side-channel cryptanalysis tolerant Rijndael symmetric block cipher architecture," in *Proc. DFT*, 2001, pp. 418–426.

[18] X. Guo, D. Mukhopadhyay, C. Jin, and R. Karri, "NREPO: Normal basis recomputing with permuted operands," in *Proc. HOST*, Arlington, VA, USA, 2014, pp. 118–123.

[19] K. Wu, R. Karri, G. Kuznetsov, and M. Goessel, "Low cost concurrent error detection for the Advanced Encryption Standard," in *Proc. Test Conf.*, Charlotte, NC, USA, 2004, pp. 1242–1248.

[20] A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A compact Rijndael hardware architecture with S-box optimization," in *Proc. ASIACRYPT*, 2001, pp. 239–254.

[21] D. Canright, "A very compact S-box for AES," in *Proc. CHES*, 2005, pp. 441–455.
[22] Y. Nogami, K. Nekado, T. Toyota, N. Hongo, and Y. Morikawa, "Mixed bases for efficient inversion in $F((2^2)^2)^2$ and conversion matrices of SubBytes of AES," in *Proc. CHES*, 2010, pp. 234–247.
[23] R. Ueno, N. Homma, Y. Sugawara, Y. Nogami, and T. Aoki, "Highly efficient $GF(2^8)$ inversion circuit based on redundant GF arithmetic and its application to AES design," in *Proc. Cryptograph. Hardw. Embedded Syst.*, 2015, pp. 63–80.
[24] G. Canivet *et al.*, "Glitch and laser fault attacks onto a secure AES implementation on a SRAM-based FPGA," *J. Cryptol.*, vol. 24, no. 2, pp. 247–268, 2011.
[25] T. Itoh and S. Tsujii, "A fast algorithm for computing multiplicative inverses in $GF(2^m)$ using normal bases," *Inf. Comput.*, vol. 78, no. 3, pp. 171–177, 1988.
[26] N. F. Ghalaty, B. Yuce, and P. Schaumont, "Analyzing the efficiency of biased-fault based attacks," *Embedded Syst. Lett.*, vol. 8, no. 2, pp. 33–36, Jun. 2016.
[27] N. F. Ghalaty, B. Yuce, M. M. I. Taha, and P. Schaumont, "Differential fault intensity analysis," in *Proc. FDTC*, 2014, pp. 49–58.
[28] S. Patranabis, A. Chakraborty, P. H. Nguyen, and D. Mukhopadhyay, "A biased fault attack on the time redundancy countermeasure for AES," in *Proc. COSADE*, 2015, pp. 189–203.
[29] S. Patranabis, A. Chakraborty, D. Mukhopadhyay, and P. Ha Nguyen. (2015). *Using State Space Encoding to Counter Biased Fault Attacks on AES Countermeasures*. [Online]. Available: https://eprint.iacr.org/2015/806.pdf

Mehran Mozaffari Kermani (S'00–M'11–SM'16) received the B.Sc. degree from the University of Tehran, Tehran, Iran, and the M.E.Sc. and Ph.D. degrees from the University of Western Ontario, London, ON, Canada, in 2007 and 2011, respectively.

He joined the Advanced Micro Devices, Markham, ON, Canada, as a Senior ASIC/Layout Designer, integrating sophisticated security/cryptographic capabilities into accelerated processing. In 2012, he joined the Electrical Engineering Department, Princeton University, Princeton, NJ, USA, as an NSERC Post-Doctoral Research Fellow. From 2013 to 2017, he was an Assistant Professor with the Rochester Institute of Technology, Rochester, NY, USA. In 2017, he has joined the Computer Science and Engineering Department, University of South Florida, Tampa, FL, USA.

Dr. Mozaffari Kermani was a recipient of the prestigious Natural Sciences and Engineering Research Council of Canada Post-Doctoral Research Fellowship in 2011 and the Texas Instruments Faculty Award (Douglas Harvey) in 2014. He is currently serving as an Associate Editor for the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS, the *ACM Transactions on Embedded Computing Systems*, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—I. He has been the Guest Editor for the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, the IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, and the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING for special issues on security. He has been the TPC member for a number of conferences including HOST (Publications Chair), Design Automation Conference, Design Automation and Test in Europe Conference, RFIDSec, LightSec, International Workshop on the Arithmetic of Finite Fields, Fault Diagnosis and Tolerance in Cryptography Workshop, and International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems.

Amir Jalali received the B.Sc. degree in computer engineering from Shahid Beheshti University, Tehran, Iran, in 2009 and the M.Sc. degree in computer engineering from the Department of Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, in 2012. He is currently pursuing the Ph.D. degree in computer engineering with the Department of Computer, Electrical Engineering and Computer Science, Florida Atlantic University, Boca Raton, FL, USA.

He is an I-SENSE Researcher with Florida Atlantic University. His current research interests include efficient software implementation of elliptic curve cryptography and post-quantum cryptography.

Reza Azarderakhsh (M'12) received the B.Sc. degree in electrical and electronic engineering and the M.Sc. degree in computer engineering from the Sharif University of Technology, Tehran, Iran, in 2002 and 2005, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Western Ontario, London, ON, Canada, in 2011.

He joined the Department of Electrical and Computer Engineering, University of Western Ontario, as a Limited Duties Instructor, in 2011, where he has been an Natural Sciences and Engineering Research Council (NSERC) Post-Doctoral Research Fellow with the Center for Applied Cryptographic Research and the Department of Combinatorics and Optimization. He is currently with the Department of Computer and Electrical Engineering and Computer Science and is an I-SENSE Fellow, Florida Atlantic University, Boca Raton, FL, USA. His current research interests include finite field and its application, elliptic curve cryptography, and pairing-based cryptography.

Dr. Azarderakhsh was a recipient of the prestigious NSERC of Canada Post-Doctoral Research Fellowship in 2012. He is currently serving as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS I. He was the Guest Editor for the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING for the special issue of Emerging Embedded and Cyber Physical System Security Challenges and Innovations, from 2016 to 2017. He was also the Guest Editor for the IEEE TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS for the special issue of Emerging Security Trends for Biomedical Computations, Devices, and Infrastructures, from 2015 to 2016.

Jiafeng Xie (M'15) received the B.E. degree in measurement and control technology and instrumentation from Yanshan University, Qinhuangdao, China, in 2006, the M.E. degree in control science and engineering from Central South University, Changsha, China, in 2010, and the Ph.D. degree in electrical engineering from the University of Pittsburgh, Pittsburgh, PA, USA, in 2014.

He is currently an Assistant Professor with Department of Electrical Engineering, Wright State University, Dayton, OH, USA. His current research interests include very large scale integration (VLSI) cryptographic circuits design, intelligent system fault detection, hardware security, VLSI, and signal/image processing systems.

Kim-Kwang Raymond Choo (SM'15) received the Ph.D. degree in information security from the Queensland University of Technology, Brisbane, QLD, Australia, in 2006.

He holds the Cloud Technology Endowed Professorship with the University of Texas at San Antonio, San Antonio, TX, USA. He is an Associate Professor with the University of South Australia, Adelaide, SA, Australia, and a Guest Professor with the China University of Geosciences, Wuhan, China.

Dr. Choo was a recipient of various awards, including the ESORICS 2015 Best Paper Award, the Winning Team of the Germany's University of Erlangen-Nuremberg (FAU) Digital Forensics Research Challenge in 2015, the 2014 Highly Commended Award by the Australia New Zealand Policing Advisory Agency, the Fulbright Scholarship in 2009, the 2008 Australia Day Achievement Medallion, and the British Computer Society's Wilkes Award in 2008. He is a fellow of the Australian Computer Society.