IEEE TRANSACTIONS ON
EMERGING TOPICS
IN COMPUTING

Received 9 February 2020; revised 2 July 2020; accepted 26 September 2020.
Date of publication 1 October 2020; date of current version 4 March 2022.

Digital Object Identifier 10.1109/TETC.2020.3027789

Hardware Constructions for Lightweight Cryptographic Block Cipher QARMA With Error Detection Mechanisms

JASMIN KAUR, (Student Member, IEEE), MEHRAN MOZAFFARI KERMANI, (Senior Member, IEEE), AND REZA AZARDERAKHSH, (Member, IEEE)

Jasmin Kaur and Mehran Mozaffari Kermani are with the Department of Computer Science and Engineering, University of South Florida, Tampa FL 33620, USA
Reza Azarderakhsh is with the Department of Computer and Electrical Engineering and Computer Science, I-SENSE Fellow, Florida Atlantic University, Boca Raton FL 33431, USA
CORRESPONDING AUTHOR: M. MOZAFFARI-KERMANI (mehran2@usf.edu)

ABSTRACT The cryptographic algorithm QARMA is a family of lightweight tweakable block ciphers targeted at applications such as memory encryption and construction of keyed hash functions. Utilizing lightweight security in hardware has the advantage of adopting the mechanisms to battery-constrained usage models including implantable and wearable medical devices. This lightweight block cipher utilizes a substitution permutation network (SPN) which is inspired by block ciphers such as PRINCE, MANTIS, and MIDORI. Moreover, it uses a three-round Even-Mansour scheme instead of the FX-construction, with its central permutation being non-involutory and keyed. In this article, we introduce error detection schemes on variations of QARMA, namely QARMA-64 and QARMA-128, which to the best of authors' knowledge, have not been presented to date. We present our derivations for the logic-gate-based implementation, following which, we present the derivations for signature-based and interleaved signature-based schemes for the LUT-based approach. The presented, new signature-based error detection schemes, including cyclic redundancy check (CRC), are provided for the compact, involutory, and optimized S-box. Besides, recomputations through encoding the operands allow for the architectures to counter both transient and permanent faults. Also, the schemes are benchmarked on a field-programmable gate array (FPGA) hardware platform, where performance and implementation metrics show acceptable overheads and degradations. The proposed schemes are aimed to make the implementations of this lightweight tweakable block cipher more reliable.

INDEX TERMS Error detection, field-programmable gate array (FPGA), lightweight tweakable block cipher, QARMA

I. INTRODUCTION

Lightweight cryptography is crucial to modern deeply-embedded systems, i.e., those embedded in human bodies and objects, including area and power/energy-constrained implantable and wearable medical devices, smart buildings, and the Internet of nano-Things. Such variant of cryptography is an optimized approach for secure embedded systems, especially for those applications and devices that are required to have a low area footprints and power/energy consumption levels such as RFID tags, wireless nano-sensors, and the like [1]. Research on lightweight cryptography has resulted in block ciphers which aim to achieve high- security levels like those of the Advanced Encryption Standard (AES), low energy consumption, and low hardware complexity for constrained devices [2]. On the other hand, tweakable block ciphers [3] focus primarily on designs of encryption modes and hash functions for specific applications such as disk encryption [4], and memory encryption [5]. Nevertheless, without taking proper measures, the constructions and implementations of these block ciphers could be vulnerable to security attacks [6], [7], making them unsafe. We note that, however, the security attacks such as meet-in-the-middle attack differ from implementation attacks commonly known as side-channel attacks. The focus of our paper is to propose error detection schemes for QARMA.

QARMA [8] is a family of lightweight tweakable block ciphers with a bricklayer substitution permutation network

2168-6750 © 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE

Authorized licensed use limited to: Florida Atlantic University. Downloaded on March 24,2021 at 00:11:20 UTC from IEEE Xplore. Restrictions apply.

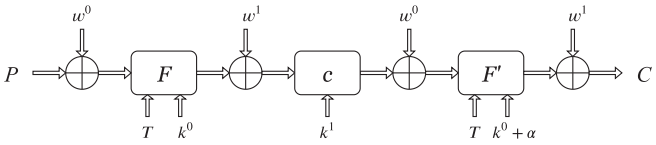

FIGURE 1. The overall scheme of QARMA. The first permutation (F) and the third permutation (F') are functionally inverse and are parameterized by a tweak (T). The central permutation (c) is inverted using a key transformation.

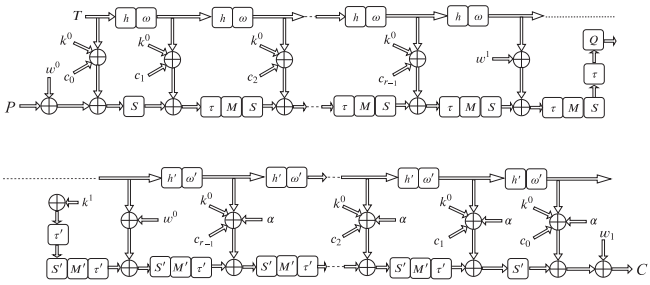

FIGURE 2. A detailed structure of QARMA having r rounds. It shows the *three-round Even-Mansour scheme* with the keyed pseudo-reflector central construction.

(SPN) that adopts a *three-round Even-Mansour scheme* keyed with a pseudo-reflector. The diffusion layer uses a family of Almost-(Maximum Distance Separable: MDS) matrices defined over a ring with zero divisors. The heuristics for the S-boxes are also chosen such that it has a minimum delay. QARMA can choose from three different S-boxes depending upon the application requirements, and has two variants, namely QARMA-64 and QARMA-128.QARMA is mainly used for memory encryption, nonce rotation, tweak compression, hash generation, and has already been deployed for pointer authentication by ARM [9]. Its small area, low latency, and low power consumption also make it suitable for use in parallelism (pipelining or unrolling architectures).

Proposing reliable error detection schemes for lightweight block ciphers have been a prominent research topic in lightweight cryptography [10]–[16]. Signature-based schemes have been adopted in works [10], [14], [16] for detection of natural and malicious faults, while recomputing with encoded operands schemes in [15] are used for high efficiency and error coverage. QARMA, shown in Figure 1, takes its cryptographic properties from the block ciphers PRINCE, MIDORI and MANTIS [8]. Even though QARMA claims to have acceptable security margins, however, its implementations can be prone to natural defects, malicious faults aiming at erroneous outputs, and fault attacks. Therefore, by implementing the proposed hardware architectures for error detection, QARMA's hardware architectures can be augmented for fault detection.

In this paper, the logic-gate-based and look-up table (LUT)-based error detection schemes are presented for the hardware implementations of QARMA, as no prior work has been introduced on this to the best of authors' knowledge. These schemes are designed to provide high error coverage, low overhead, and resistance against faults as well as to augment the original constructions with VLSI reliability. They impracticably do not undermine the performance and implementation metrics of the original design. First, the derivations for the logic-gate-based implementation, following which, the derivations for signature-based and interleaved signature-based schemes for the LUT-based approach are presented. These signature-based schemes are tailored for signature (for single faults), interleaved signature (for burst/multiple faults), cyclic redundancy check (CRC), and architecture-oblivious (recomputed with encoded operands) constructions. The proposed approaches can be adopted to similar cryptographic algorithms with sub-blocks presented in this paper, making them not just confined to QARMA cryptographic algorithm. Specifically, NIST candidates for lightweight cryptography can take advantage of the proposed constructions. We also benchmark the proposed error detection architectures on field-programmable gate array (FPGA) hardware platform to confirm the achieved objectives, noting that the proposed approaches are oblivious of the platform and we expect similar acceptable overheads on application-specific integrated circuit (ASIC).

The paper is organized as follows: In Section II, preliminaries describing the functionality of QARMA are presented. In Section III, we present the proposed error detection schemes. In Section IV, the error coverage assessments and FPGA implementations are presented. In Section V, finally, the conclusions are made.

II. PRELIMINARIES

The overall scheme of QARMA is shown in Figure 1, while its structure is shown in Figure 2.

The encryption process is a sequence of operations on the $n = 64$-bit or $n = 128$-bit internal state (IS), represented as arrays of $16m$-bit cells, combined with a tweak and a key. The tweak used is n-bit long, the key is $2n$-bit long, and the rounds are calculated as $2r + 2$ for a given r. The permutations are parameterized using a core key and the round keys are derived using a whitening key. The $2n = 32m$-bit master key is split into $w^0||k^0$, where w^0 and k^0 are the $16m$-bit whitening and core keys, respectively. For encryption, $k^1 = k^0$ and $w^1 = o(w^0)$, where $o(x)$ is an orthomorphism defined as $o(x) := (x \ggg 1) + (x \gg (16m - 1))$. Here, the $\gg$ symbol and the $\ggg$ symbol denote a right shift and a right circular shift of the register bits, respectively.

The forward round function F of QARMA has the following four operations:

1) AddingRoundTweaky (K): For this operation, the ith n-bit round key K_i is XORed with the state IS, the tweak (t), and the round constant (c_i). The permutation $h = [6, 5, 14, 15, 0, 1, 2, 3, 7, 12, 13, 4, 8, 9, 10, 11]$ is used to permute the cells of the tweak, and an LFSR ω is used to update the tweak.
2) ShuffleCells (τ): For this operation, $(\tau(IS))_i = s_{\tau(i)}$, where $0 \leqslant i \leqslant 15$. It uses the cell permutation $\tau = [0, 11, 6, 13, 10, 1, 12, 7, 5, 14, 3, 8, 15, 4, 9, 2]$ on the IS.
3) MixColumns (M): In this operation, each column of the IS array is multiplied by the matrix M, i.e., $IS = IS \cdot M$,

Authorized licensed use limited to: Florida Atlantic University. Downloaded on March 24,2024 at 00:11:20 UTC from IEEE Xplore. Restrictions apply.

TABLE 1. Signature $(\hat{p_0})$ and interleaved signature $(\hat{p_1}, \hat{p_2})$ of QARMA's involutive S-box σ (in hexadecimal form).

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$\sigma(x)$	A	D	E	6	F	7	3	5	9	8	0	C	B	1	2	4
$\hat{p_0}$	0	1	1	0	0	1	0	0	0	1	0	0	1	1	1	1
$(\hat{p_1}, \hat{p_2})$	(0,0)	(0,1)	(1,0)	(1,1)	(0,0)	(0,1)	(1,1)	(0,0)	(1,1)	(0,1)	(0,0)	(1,1)	(1,0)	(1,0)	(0,1)	(1,0)

where M is involutory. The multiplication of an element of the array with ρ^i is a left circular rotation of the element by i bits.

4) SubCells (S): For this operation, the IS is updated using the chosen m-bit S-box as $s_i \leftarrow \sigma(s_i)$, where $0 \leqslant i \leqslant 15$ and $m = 4$ or $m = 8$. For our implementation, the chosen optimized involutory 8-bit bi-jective S-box $\sigma = [10, 13, 14, 6, 15, 7, 3, 5, 9, 8, 0, 12, 11, 1, 2, 4]$, is applied in parallel on each byte of the IS of QARMA-128.

The backward round function F' is the inverse of the forward round function. The central construction c consists of a forward round, a backward round, a Pseudo-Reflector, and a key (derived from the whitening key).

III. PROPOSED ERROR DETECTION ARCHITECTURES

In this section, the efficient and overhead-aware error detection schemes for QARMA are presented. In the proposed architectures, we have based our schemes on different tiers of approaches requiring wide range of reliability, overhead, and obliviousness of the architectures.

A. PROPOSED SIGNATURE-BASED SCHEMES FOR THE OPTIMIZED S-BOX

For the SubCells operation of encryption process in QARMA-128, each sub-byte of the IS is passed through the instances of the 8-bit S-box which is constructed by placing two 4-bit S-boxes in parallel. The input and output bits of the S-box are wired in an asymmetric manner, where the output bits are cyclically rotated and fed to the next instance of the S-box. In this paper, we focus mainly on the homogeneous S-box σ used in QARMA which is optimal, compact, and involutory. For hardware implementations of the S-box σ, we propose the following approaches, which are applicable to variants of the S-box, i.e., the logic-gate-based and the LUT-based. The first approach has the advantage of having low area and low power consumption, while the second approach has the advantage of having higher error coverage but at the cost of area and power consumption. We also present a CRC-3 construction to show that the proposed work can be tailored based on the error coverage needs and overhead to be tolerated. Since the 8-bit S-box of QARMA-128 is bijective, the equations are derived only for the 4-bit variant.

We first present our derivations for the logic-gate-based implementation of the S-Box σ. Next, we present the values for signature-based and interleaved signature-based schemes for the LUT-based approach in Table 1, where the one-bit signature $(\hat{p_0})$ and interleaved signatures $(\hat{p_1}, \hat{p_2})$ corresponding to each of the sixteen 4-bit elements of the S-box σ are listed. The equations of these signatures for logic-gate-based constructions are also derived below. The signatures presented in Table 1 are better suited for FPGAs, as the abundant memory units (block memories or pipelined distributed memories) can be utilized to store the signature values along with the S-Box values. While the logic-gate based constructions are better for ASICs, as the LUT-based S-boxes not preferred due to memory macros and registers being expensive in implementations. Here and throughout the paper, detailed derivations have not been presented for the sake of brevity.

Remark 1. Considering the 4-bit S-box of QARMA with the input $(\mu = (\mu_3, \mu_2, \mu_1, \mu_0))$ and output $(\nu = (\nu_3, \nu_2, \nu_1, \nu_0))$, we derive the following low-complexity equations for logic-gate-based variant:

$$\begin{aligned} \nu_0 &= \mu_2(\bar{\mu_1} \vee \bar{\mu_3}) \vee \bar{\mu_1}(\mu_3 + \mu_0), \\ \nu_1 &= \bar{\mu_0}(\bar{\mu_3} \vee \mu_2) \vee \bar{\mu_3}(\mu_1 + \mu_2), \\ \nu_2 &= \mu_0(\bar{\mu_3} \vee \mu_1) \vee \bar{\mu_3}(\mu_1 + \mu_2), \\ \nu_3 &= \bar{\mu_1}(\bar{\mu_2} \vee \bar{\mu_0}) \vee \bar{\mu_2}(\bar{\mu_3} + \mu_0). \end{aligned} \tag{1}$$

Here and throughout the paper, the symbols $\vee$, $+$, and the *bar* represent OR, XOR, and NOT operations, respectively.

Theorem 1. Considering the 4-bit S-box of QARMA with the input as $(\mu = (\mu_3, \mu_2, \mu_1, \mu_0))$ and the output as $(\nu = (\nu_3, \nu_2, \nu_1, \nu_0))$, the one-bit signature is derived as follows:

$$\hat{p_0} = \mu_3\mu_2 \vee \bar{\mu_1}\mu_0 \vee \bar{\mu_3}\bar{\mu_2}\mu_1\bar{\mu_0}, \tag{2}$$

Proof. The predicted signatures corresponding to each element of the given S-box are computed by modulo-2 addition of all the output bits corresponding to the input bits. Hence, using (1) we get, $\hat{p_0} = \mu_3\mu_2\bar{\mu_1}\bar{\mu_0} \vee \mu_3\mu_2\bar{\mu_1}\mu_0 \vee \mu_3\mu_2\mu_1\mu_0 \vee \mu_3\mu_2\mu_1\bar{\mu_0} \vee \bar{\mu_3}\bar{\mu_2}\bar{\mu_1}\mu_0 \vee \bar{\mu_3}\mu_2\bar{\mu_1}\mu_0 \vee \mu_3\mu_2\bar{\mu_1}\mu_0 \vee \mu_3\bar{\mu_2}\bar{\mu_1}\mu_0 \vee \bar{\mu_3}\bar{\mu_2}\mu_1\bar{\mu_0}$, which on further simplification gives us the final equation as (2). □

Theorem 2. Considering the 4-bit S-box of QARMA with the input as $(\mu = (\mu_3, \mu_2, \mu_1, \mu_0))$ and the output as $(\nu = (\nu_3, \nu_2, \nu_1, \nu_0))$ of the S-box, the interleaved signatures are derived as follows:

$$\hat{p_1} = \bar{\mu_3}\bar{\mu_2}\mu_1 \vee (\mu_3 + \mu_1)\bar{\mu_0} \vee \mu_3\mu_1\mu_0 \vee \mu_3\mu_2\bar{\mu_1}, \tag{3}$$

$$\hat{p_2} = \bar{\mu_2}\mu_0 \vee \mu_3\bar{\mu_2}\bar{\mu_1} \vee \bar{\mu_3}\bar{\mu_1}\mu_0 \vee \mu_2\mu_1\bar{\mu_0}. \tag{4}$$

Proof. The interleaved parities are computed by the separate modulo-2 addition of odd bits and even bits of the

Authorized licensed use limited to: Florida Atlantic University. Downloaded on March 24,2024 at 00:11:20 UTC from IEEE Xplore. Restrictions apply.

TABLE 2 **CRC-3 signatures $\hat{p_3}, \hat{p_4}, \hat{p_5}$ of QARMA's involutive S-box σ (in hexadecimal form).**

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$\sigma(x)$	A	D	E	6	F	7	3	5	9	8	0	C	B	1	2	4
$\hat{p_3}$	1	0	1	0	0	1	1	1	0	1	0	1	0	1	0	0
$\hat{p_4}$	0	1	0	1	0	1	1	0	1	1	0	1	0	0	1	0
$\hat{p_5}$	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	1

sixteen 4-bit entries of the S-box. Using our derived detailed formulations in (1), the equation for odd-bit parity is formulated as $\hat{p_1} = \mu_3\mu_2\bar{\mu_1}\bar{\mu_0} \vee \mu_3\mu_2\bar{\mu_1}\mu_0 \vee \mu_3\mu_2\mu_1\ \mu_0 \vee \mu_3\bar{\mu_2}\ \mu_1\mu_0 \vee \bar{\mu_3}\bar{\mu_2}\mu_1\bar{\mu_0}\ \vee\bar{\mu_3}\mu_2\mu_1\bar{\mu_0} \vee \mu_3\bar{\mu_2}\bar{\mu_1}\bar{\mu_0} \vee \mu_3\bar{\mu_2}\mu_1\mu_0$, which gives us the equation $\hat{p_1} = (\nu_0 + \nu_2) = \bar{\mu_3}\bar{\mu_2}\mu_1 \vee \bar{\mu_3}\mu_1\bar{\mu_0} \vee \mu_3\mu_1\mu_0 \vee \mu_3\mu_2\bar{\mu_1} \vee \mu_3\bar{\mu_1}\bar{\mu_0}$. On further simplification, we get the equation in (3), i.e., $\hat{p_1} = \bar{\mu_3}\bar{\mu_2}\mu_1 \vee (\mu_3 + \mu_1)\bar{\mu_0} \vee \mu_3\mu_1\mu_0 \vee \mu_3\mu_2\bar{\mu_1}$. Similarly, the even-bit parity is derived as, $\hat{p_2} = (\nu_1 + \nu_3) = \bar{\mu_3}\bar{\mu_2}\bar{\mu_1}\mu_0 \vee \bar{\mu_3}\bar{\mu_2}\ \mu_1\mu_0 \vee \bar{\mu_3}\mu_2\bar{\mu_1}\mu_0 \vee \bar{\mu_3}\mu_2\ \mu_1\bar{\mu_0} \vee \mu_3\mu_2\mu_1\bar{\mu_0} \vee \mu_3\bar{\mu_2}\bar{\mu_1}\bar{\mu_0}\ \vee\mu_3\bar{\mu_2}\bar{\mu_1}\mu_0 \vee \mu_3\bar{\mu_2}\mu_1\mu_0$, which gives us the equation in (4), i.e., $\hat{p_2} = \bar{\mu_2}\mu_0 \vee \mu_3\bar{\mu_2}\bar{\mu_1} \vee\bar{\mu_3}\bar{\mu_1}\mu_0\vee\ \mu_2\mu_1\bar{\mu_0}$. □

For our other approach, we present a CRC-3 scheme to account for the needs for higher error coverage at the expense of higher overhead. The following theorem derives signatures $\hat{p_3}$,$\hat{p_4}$, and $\hat{p_5}$ for the presented CRC-3 scheme.

Theorem 3. Considering the 4-bit S-box of QARMA with the input as $(\mu = (\mu_3, \mu_2, \mu_1, \mu_0))$ and the output as $(\nu = (\nu_3, \nu_2, \nu_1, \nu_0))$, we get the CRC-3 signatures as:

$$\hat{p_3} = \bar{\mu_2}(\mu_3 + \bar{\mu_0}) \vee \mu_2(\bar{\mu_3}\mu_1 \vee \bar{\mu_1}\mu_0), \tag{5}$$

$$\hat{p_4} = \bar{\mu_2}\mu_0 \vee \bar{\mu_1}(\mu_3\bar{\mu_2} \vee \bar{\mu_3}\mu_0) \vee \mu_2\mu_1\bar{\mu_0}, \tag{6}$$

$$\hat{p_5} = \mu_0(\bar{\mu_3} \vee \mu_1) \vee \bar{\mu_3}(\mu_2 + \mu_1). \tag{7}$$

Proof. For CRC-3, we perform modular reduction using the CRC-3 polynomial $g(x) = x^3 + x + 1$. Using our derivations, we get the equation for the signature as $\nu_3(x + 1) + \nu_2x^2 + \nu_1x + \nu_0$, where the coefficients terms of x^0, x^1, and x^2 correspond to the individual signature bits $\hat{p_3}$,$\hat{p_4}$, and $\hat{p_5}$, respectively. The equation on further simplification gives us the final equation as $\nu_2x^2 + (\nu_1 + \nu_3)x + (\nu_0 + \nu_3)$ for the given output vectors $(\nu = (\nu_3, \nu_2, \nu_1, \nu_0))$. By substituting the output vectors with corresponding input vectors $(\mu = (\mu_3, \mu_2, \mu_1, \mu_0))$, we get the simplified equations of the signature bits as shown in (5), (6), and (7), respectively. □

We have also depicted the CRC-3 signatures in Table 2 based on formulations (5), (6), and (7), specifically for LUT-based S-boxes where these values can be stored in memories along with the S-box output bits. The proposed architecture-dependent scheme for one instance of the 8-bit involutive S-box of QARMA-128 is shown in Figure 3.

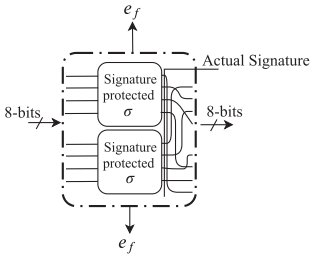

FIGURE 3. **The proposed architecture-dependent scheme for one instance of the 8-bit involutive S-box of QARMA-128. The two e_f depict the error indication flags for the proposed schemes.**

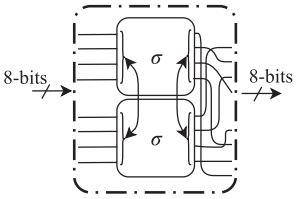

FIGURE 4. **The proposed architecture-oblivious scheme for 8-bit involutive S-box of QARMA-128 as implemented on one of the instances. Here, the input and output bits are swapped.**

B. LOW-AREA/POWER STRUCTURE-OBLIVIOUS SCHEME FOR THE S-BOX

For the 8-bit bijective S-box in QARMA-128 described above, we propose architecture-oblivious error detection scheme, realized through recomputing with encoded operands, to give the designers freedom in choosing the S-box constructions. In this approach, as shown in Figure 4, we swap the input nibbles to the 8-bit S-box in QARMA-128, i.e., the first four inputs go to the second 4-bit S-box and the next four inputs are fed to the first 4-bit S-box. The correct result is obtained by swapping the output bits of the S-box, which then in comparison with the actual output detects transient and permanent faults. This proposed scheme does not affect the algorithmic security of QARMA-128, since the error detection is performed via re-computation, hence the original structure of QARMA-128 is protected. The results are presented in Section IV.

C. PROPOSED SCHEMES FOR OTHER OPERATIONS

1) SCHEME FOR MIXCOLUMNS OPERATION OF QARMA

QARMA can choose from several matrices for its MixColumns operation, in which each column of *IS* is multiplied by an involutory matrix M. The multiplication of an element of *IS* by an element ρ^i of the matrix M is a left circular shift of the bits of the *IS* element by i bits. For example, in QARMA-128, the following matrix with 8-bit elements is used for array multiplication:

$$M_8 = Q_8 = \begin{pmatrix} 0 & \rho & \rho^4 & \rho^5 \\ \rho^5 & 0 & \rho & \rho^4 \\ \rho^4 & \rho^5 & 0 & \rho \\ \rho & \rho^4 & \rho^5 & 0 \end{pmatrix}.$$

Authorized licensed use limited to: Florida Atlantic University. Downloaded on March 24,2024 at 00:11:20 UTC from IEEE Xplore. Restrictions apply.

Remark 2. The column-wise signatures for the outputs of MixColumns are equal to modulo-2 addition of columns of the state input matrix. This results in an efficient and low-complexity error detection scheme.

2) SCHEME FOR ADDROUNDKEY OPERATION OF QARMA

In this operation, the round key K is XORed with the IS, a tweak τ, and round constant c. The tweak is permuted using the permutation h and only the cells with indices [0,1,3,4,8,11,13] are updated using LFSR ω. This results in a simple yet efficient error detection scheme through modulo-2 addition of the state and the key/tweak.

3) SCHEME FOR DECRYPTION OF QARMA

The decryption operation in QARMA is the same as encryption operation, except that $k_0 + \alpha$ is used as the core key, $k^1 = Q.k^0$, the whitening keys w^0 and w^1 are swapped, and $Q{\cdot}t$ is used as the tweakey. Hence, the same error detection schemes that are used for encryption can be implemented for decryption as well.

IV. ERROR COVERAGE AND FPGA BENCHMARK

Our proposed error detection schemes are designed to detect transient faults and permanent faults which can occur due to malicious or natural faults. An attacker may not be successful at flipping exactly one bit to collect sensitive information due to technological constraints. Therefore, in addition to single faults, we need to consider schemes that are able to detect multiple stuck-at faults (stuck-at 0 and stuck-at 1). Moreover, parity, interleaved parity, CRC-3, and recomputing with encoded operands can be combined to achieve high error coverage. Such schemes are needed for detecting random faults, natural faults, and to also provide security against intelligent attackers who may rely on other vulnerabilities of the design (side-channel attacks). In practice, the attacker is interested in using as few faults as possible (preferably single faults with different intensities) to minimize the effort. Previous works 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 [17]. In addition, fault categories presented 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 [17].

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. 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. The proposed approaches are for VLSI reliability and also make fault attacks more difficult; however, it is possible for the attacker to inject faults to the error detection architecture. In such cases, the attacks and technologies to mount them would be more sophisticated, yet this is a possibility. Hardening the error detection architecture through larger parallel transistor-based logic is a viable remedy in this case. Through simulations, the error coverage for stuck-at faults was evaluated for QARMA, specifically for the proposed CRC-3 and the recomputing with encoded operands scheme. Single and multiple stuck-at faults were considered to cover both natural (permanent) and malicious (transient) fault injections. Multiple stuck-at faults were injected in the inputs of the S-Box to generate faulty outputs by forcing the input bits to be either 0 or 1. The faulty outputs were then compared to the predicted outputs and the error indication flags of the proposed schemes were monitored. For the single stuck-at faults, the error coverage was 100 percent since the proposed schemes are designed to detect odd faults. For multiple stuck-at faults, through injecting 50,000 faults, the results for the CRC-3 and recomputing with encoded operands scheme showed error coverage of 99.46 and 99.53 percent, respectively.

TABLE 3. FPGA implementation results for the original QARMA-128 encryption and its proposed error detection scheme on Kintex UltraScale FPGA family for device -xcku035-ffva1156-2-i.

Architecture	Area ($Slices$)	Delay (ns)	Power @ 100 MHz (W)	Thr′put ($Gbps$)	Efficiency ($Gbps$/ $Slice$)
QARMA-128	418	2.795	0.525	45.80	16.38
QARMA-128 w/ one-bit signature	424 (1.44%)	2.809 (0.50%)	0.538 (2.48%)	45.57 (0.47%)	16.22 (0.98%)
QARMA-128 w/ interleaved signature	465 (11.24%)	2.910 (4.11%)	0.564 (7.43%)	43.99 (3.95%)	15.11 (7.75%)
QARMA-128 w/ CRC-3 scheme	505 (20.81%)	3.318 (18.71%)	0.579 (10.29%)	38.58 (15.77%)	11.63 (28.99%)

The benchmark of overheads of the proposed schemes, implemented for QARMA-128, are presented in this section as well. The FPGA implementation has been performed on the device -xcku035-ffva1156-2-i of Kintex UltraScale FPGA family. Xilinx Vivado version 14.1 has been used for performance and implementation metrics derivations and the RTL has been coded using Verilog. Table 3 shows the results of our implementations for the logic-gate based variant. From Table 3, we can see that the overheads of the implemented architectures are acceptable. The area overhead of one-bit signature is 1.44 percent on top of the original architecture, while the delay is higher by 0.50 percent and the power goes up by 2.48 percent. This increase is due to additional clock-cycles that are needed to compute the predicted and actual parities which are then compared to original signatures for error flags. The table also tabulates the throughput (bits at the output every second) and the efficiency (which is throughput over area) for the architectures. The performance and implementation metrics for the proposed implemented CRC-3 scheme are also presented in this table. The overheads for the LUT-based variant of our proposed schemes are expected to be similar to the overheads mentioned above as the signature values are stored in memory along with the contents of the S-Box σ, and does not require additional circuitry.

Authorized licensed use limited to: Florida Atlantic University. Downloaded on March 24,2024 at 00:11:20 UTC from IEEE Xplore. Restrictions apply.

TABLE 4. Implementation results for the original QARMA-128 encryption and its proposed recomputing with encoded operands scheme on Kintex UltraScale FPGA family for device -xcku035-ffva1156-2-i.

Architecture	Area (*Slices*)	Delay (*ns*)	Power @ 100 MHz (*W*)
QARMA-128	543	3.611	0.522
QARMA-128 w/ Recomputing	653 (20.26%)	4.078 (12.93%)	0.611 (17.05%)

In absence of any compensation, the total time of recomputing architectures that do not embed throughput alleviation approaches is not acceptable. This drastic deterioration of the throughput can be improved by incorporating subpipelining. The design throughput will be close to the original architecture as subpipelining increases the frequency. Table 4 shows the overhead results of recomputing with encoded operands scheme depicted in Figure 4. For the recomputing with encoded operands scheme, the output is calculated twice, hence, giving us the overheads for the area, delay, and power consumption as 20.26, 12.93, and 17.05 percent, respectively. Due to runtime retiming of the implemented circuit on FPGA, the presented overheads can slightly deviate from the expected overheads for this scheme. Throughput can be improved by using subpipelining, i.e., by putting a register array between S-boxes to store both the predicted and the actual output for comparison, as explained above. These proposed schemes or combinations of the results in Tables 3 and 4 can be used to achieve the reliability requirements and the required implementation and performance metrics to tailor for deeply-embedded systems.

There has not been any prior work done on this type of error detection methods for QARMA to the best of our knowledge. However, for qualitative comparison to verify that the overheads incurred are acceptable, let us go over some case studies. The work in [15] presented signature-based fault diagnosis for cryptographic block ciphers LED and HIGHT, obtaining a combined area and delay overhead of 21.9 and 31.9 percent for LED and HIGHT, respectively. Additionally, the authors in [16] propose efficient error detection architectures of hash-counter-hash tweakable enciphering schemes, obtaining a combined area and throughput overhead of less than 13.5 percent. Such prior work on classical cryptography verifies that the proposed error detection architectures obtain acceptable overhead.

V. CONCLUSION

In this paper, we have presented signature-based error detection schemes for the lightweight block cipher QARMA to provide fault detection and VLSI reliability due to natural defects, e.g., single-event upsets (SEUs). We note that we have proposed both architecture-dependent low-complexity approaches as well as structure-oblivious schemes that are applicable to different S-boxes implementations. For the S-boxes of QARMA, we have derived and implemented both LUT-based and logic gate variants, where the presented schemes can be tailored to the reliability and overheard requirements. We note that on FPGAs, memory units (block memories or pipelined distributed memories) are abundant but on ASICs, memory macros and registers are expensive and LUT-based S-boxes are thus not preferred. Through FPGA implementations using Xilinx Kintex UltraScale family, it has been shown that the overheads of the proposed architectures are acceptable for resource-constrained applications. We would like to emphasize that the proposed approaches are mainly aimed at VLSI reliability but we envision that they make some fault attacks more difficult to mount.

ACKNOWLEDGMENTS

This work was performed under the U.S. federal agency award 60NANB20D013 granted from National Institute of Standards and Technology (NIST).

REFERENCES

[1] T. Eisenbarth, S. Kumar, C. Paar, A. Poschmann, and L. Uhsadel, "A survey of lightweight-cryptography implementations," *IEEE Des. Test Comput.*, vol. 24, no. 6, pp. 522–533, Nov./Dec. 2007.

[2] G. Hatzivasilis, K. Fysarakis, I. Papaefstathiou, and C. Manifavas, "A review of lightweight block ciphers," *J. Cryptogr. Eng.*, vol. 8, no. 2, pp. 141–184, 2017.

[3] M. Liskov, R. L. Rivest, and D. Wagner, "Tweakable block ciphers," *J. Cryptol.*, vol. 24, no. 3, pp. 588–613, 2010.

[4] L. Martin, "XTS: A mode of AES for encrypting hard disks," *IEEE Secu. Privacy Mag.*, vol. 8, no. 3, pp. 68–69, May/Jun. 2010.

[5] M. Henson and S. Taylor, "Memory encryption: A survey of existing techniques," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 1–26, 2014.

[6] S. Ali, X. Guo, R. Karri, and D. Mukhopadhyay, "Fault attacks on AES and their countermeasures," in *Secure System Design Trustable Computing*. Berlin, Germany: Springer, 2016, pp. 163–208.

[7] C. Dobraunig, M. Eichlseder, T. Korak, V. Lomné, and F. Mendel, "Statistical fault attacks on nonce-based authenticated encryption schemes" in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2016, pp. 369–395.

[8] R. Avanzi, "The QARMA block cipher family," *IACR Trans. Symmetric Cryptol.*, vol. 0, no. 0, pp. 4–44, 2017.

[9] Qualcomm, "Pointer authentication on ARMv8.3," 2017. [Online] Available: https://www.qualcomm.com/media/documents/files/whitepaper-pointer-authentication-on-armv8–3.pdf

[10] M. M. Kermani and A. Reyhani-Masoleh, "A low-cost S-box for the advanced encryption standard using normal basis," in *Proc. IEEE Int. Conf. Electro/Inf. Technol.*, 2009, pp. 52–55.

[11] S. Patranabis *et al.*, "Lightweight design-for-security strategies for combined countermeasures against side-channel and fault analysis in IoT applications," *J. Hardware Syst. Secur.*, vol. 3, no. 2, pp. 103–131, 2019.

[12] M. M. Kermani and A. Reyhani-Masoleh, "Reliable hardware architectures for the third-round SHA-3 finalist Grostl benchmarked on FPGA platform," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst.*, 2011, pp. 325–331.

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

[14] M. M. Kermani and A. Reyhani-Masoleh, "A lightweight concurrent fault detection scheme for the AES S-Boxes using normal basis," in *Proc. LNCS Cryptogr. Hardware Embedded Syst.*, 2008, pp. 113–129.

[15] S. Subramanian, M. Mozaffari-Kermani, R. Azarderakhsh, and M. Nojoumian, "Reliable hardware architectures for cryptographic block ciphers LED and HIGHT," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 36, no. 10, pp. 1750–1758, Oct. 2017.

[16] M. M. Kermani, R. Azarderakhsh, A. Sarker, and A. Jalali, "Efficient and reliable error detection architectures of Hash-Counter-Hash tweakable enciphering schemes," *ACM Trans. Embedded Comput. Syst.*, vol. 17, no. 2, pp. 54:1–54:19, May 2018.

[17] S. Patranabis, A. Chakraborty, D. Mukhopadhyay, and P. H. Nguyen, "Using state space encoding to counter biased fault attacks on AES countermeasures," 2015. [Online]. Available: https://eprint.iacr.org/2015/806.pdf

Authorized licensed use limited to: Florida Atlantic University. Downloaded on March 24,2024 at 00:11:20 UTC from IEEE Xplore. Restrictions apply.