Reliable and Fault Diagnosis Architectures for Hardware and Software-Efficient Block Cipher KLEIN Benchmarked on FPGA

Anita Aghaie, *Student Member, IEEE*, Mehran Mozaffari Kermani, *Senior Member, IEEE*, and Reza Azarderakhsh, *Member, IEEE*

***Abstract*—Security-sensitive usage models, such as implantable and wearable medical devices are prone to algorithmic cryptographic attacks as well as malicious implementation attacks. The cryptographic algorithms in these cryptosystems, such as lightweight block ciphers utilized in constrained applications, face significant tradeoff between the high level of security and the efficiency of their implementation metrics. For thwarting fault analysis attacks, among effective variants of active implementation attacks, and also to detect natural faults, efficient fault diagnosis schemes for these lightweight block ciphers are essential. In this paper, for the first time, we propose error detection schemes for lightweight block cipher, KLEIN, to ameliorate its error resiliency with low hardware complexity. The proposed fault diagnosis architectures are for linear and nonlinear components of this cipher. We also consider the notion of fault space transformation for lightweight cryptography and present its potential complications. The implementation of the proposed schemes through variants of Xilinx field-programmable gate arrays and the error coverage assessed with fault injection simulations show the effectiveness of the proposed schemes with acceptable footprints.**

***Index Terms*—Fault detection, field-programmable gate array (FPGA), KLEIN, signature-based scheme.**

I. Introduction

For providing different security properties, we use cryptographic algorithms which are publicly known. To obtain the secret key, attackers can use algebraic attacks (not efficient in most cases) or side-channel attacks, such as fault attacks through observation of the corresponding erroneous outputs. In choosing the countermeasures for such attacks, not only one needs to achieve high error coverage, but the underlying countermeasures need not to undermine the implementation objectives. For instance, good choices of error detection schemes for lightweight cryptography would not impose much area/power consumption overhead to the original lightweight algorithms.

In this paper, among the lightweight block ciphers presented to date, such as PRESENT, NOEKEON, and SEA which are efficient in terms of just hardware or just software designs, we have chosen a family of block ciphers, i.e., KLEIN [1], which is designed for resource-constrained devices (efficient in software on legacy sensor platforms and at the same time, in the hardware implementations). Furthermore, we note that security analysis shows KLEIN has conservative security margin against various cryptanalyses. Similar to other substitution-permutation networks (SPNs), it has round-based structure for data processing; however, the key schedule has been realized from a Feistel network design (to prevent key recovering) which can easily boost security margin.

Cryptographic algorithms are prone to malicious fault attacks [2]–[4] and natural errors caused due to alpha particles from cosmic rays creating energetic neutrons, thermal neutrons, and the like, whose detection has been the center of the previous works, e.g., for cryptographic applications [5]–[10]. Our main contributions in this paper are summarized as follows.

1) In this paper, we have carefully chosen the proposed error detection schemes so that they are applicable to lightweight block ciphers which have similar structures as KLEIN. Specifically, we propose signature-based schemes which can be tailored not to impractically undermine the performance and implementation metrics of the original implementations. The proposed schemes can be generalized to KLEIN-like block ciphers (such as Mysterion, LED, SKINNY) and customized in terms of error coverage and overhead. For instance, the signature-based schemes for lightweight cryptography based on column-wise approach can be tailored to have parity (for single faults), interleaved parity (for burst faults), or column-wise cyclic-redundancy check. We also note that the recomputation schemes with encoded operands presented in this paper do not undermine the area and power consumption of the original implementations of lightweight block ciphers impractically, and are able to detect both transient and permanent faults in the key schedule unit. Our error detection simulations and hardware platform implementations confirm such achieved constraints.
2) Fault space transformation (FST) remedies are also presented for lightweight cryptography and the complications are assessed.
3) We benchmark the proposed architectures to assess their ability to detect transient and permanent faults by performing fault injection simulations. Moreover, we implement the proposed error detection architectures on Xilinx field-programmable gate array (FPGA) to confirm the achieved objectives.

II. Preliminaries

Data processing of KLEIN is based on SPN architecture with round transformations in which it has 12/16/20 rounds for KLEIN-64/80/96, respectively. This data process in the encryption routine consists of four operations in each round, i.e., AddRoundKey, a straightforward modulo-2 addition of round key with intermediate ciphertexts, SubNibbles, nonlinear S-boxes for each input nibble, RotateNibbles, rotating the intermediate nibbles, and MixNibbles, which uses maximum distance separable (MDS) matrices as linear

Manuscript received March 10, 2017; revised June 4, 2017; accepted August 2, 2017. Date of publication August 17, 2017; date of current version March 29, 2018. This work was supported by the U.S. Federal Agency from the U.S. Department of Commerce, National Institute of Standards and Technology under Award 60NANB16D245. This paper was recommended by Associate Editor P. Leong. *(Corresponding author: Mehran Mozaffari Kermani.)*

A. Aghaie is with the Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology, Rochester, NY 14623 USA (e-mail: aa6964@rit.edu).

M. Mozaffari Kermani is with the Department of Computer Science and Engineering, University of South Florida, Tampa, FL 33620 USA (e-mail: mehran2@usf.edu).

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

Digital Object Identifier 10.1109/TCAD.2017.2740286

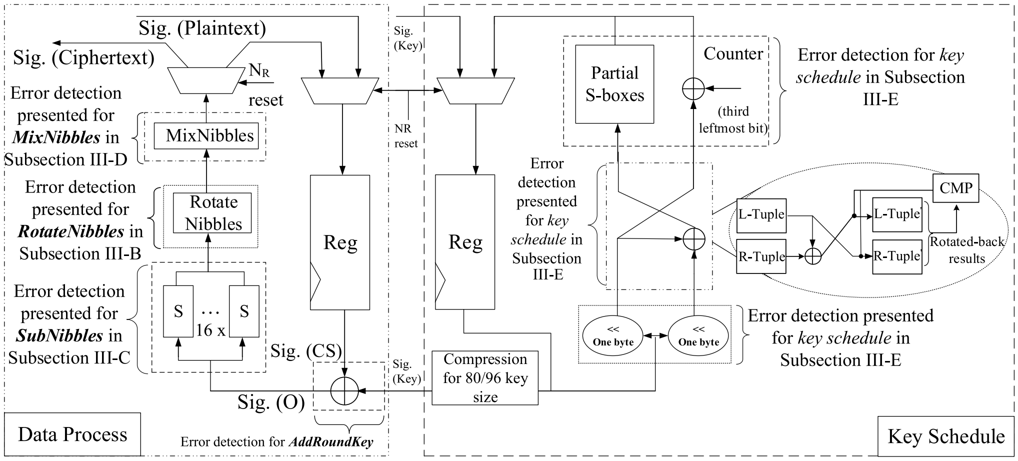
Sig. (Plaintext)
Sig. (Ciphertext)
N_R
reset
Error detection presented for *MixNibbles* in Subsection III-D
MixNibbles
Error detection presented for *RotateNibbles* in Subsection III-B
Rotate Nibbles
Error detection presented for *SubNibbles* in Subsection III-C
S
...
16 x
S
Reg
Sig. (CS)
Sig. (O)
Data Process
Error detection for *AddRoundKey*
Sig. (Key)
NR reset
Sig. (Key)
Reg
Partial S-boxes
Counter
(third leftmost bit)
Error detection for *key schedule* in Subsection III-E
Error detection presented for *key schedule* in Subsection III-E
L-Tuple
R-Tuple
L-Tuple'
R-Tuple'
CMP
Rotated-back results
<< One byte
<< One byte
Error detection presented for *key schedule* in Subsection III-E
Compression for 80/96 key size
Key Schedule

Fig. 1. Proposed error detection architecture for KLEIN-64/80/96 encryption.

TABLE I
PARITY (PAR.) AND INTERLEAVED PARITY (IPAR.) BITS OF 4-BIT INVOLUTIVE S-BOX IN HEXADECIMAL FORM

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Par.	7 (1)	4 (1)	A (0)	9 (0)	1 (1)	F (0)	B (1)	0 (0)	C (0)	3 (0)	2 (1)	6 (0)	8 (1)	E (1)	D (1)	5 (0)
IPar.	(10)	(01)	(00)	(11)	(01)	(00)	(01)	(00)	(11)	(11)	(10)	(11)	(10)	(01)	(10)	(00)

function over the nibbles. All of these operations are illustrated separately in details on the left side of Fig. 1 denoted as the data process part.

The KLEIN key schedule outputs the round-keys, e.g., in the case of 64-bit key, the initial subkey (sk^1) is divided to two 4-byte tuples ($sk^1_{0+j}||sk^1_{1+j}||sk^1_{2+j}||sk^1_{3+j}$), $j = 0, 4$. In the first step, each 4-byte unit is shifted one byte to the left cyclically as ($sk^1_{1+j}||sk^1_{2+j}||sk^1_{3+j}||sk^1_{0+j}$), $j = 0, 4$. In the second step, the right tuple becomes the next left tuple, and the remaining 4-byte tuple is modulo-2 added with the right one and in the last step, the round counter i is modulo-2 added with the third-byte in the left tuple. Moreover, in the right tuple, the second and the third bytes are substituted by the KLEIN S-box as shown in the right side of Fig. 1 in the key schedule part.

III. PROPOSED FAULT DETECTION ARCHITECTURES

In this section, through signature-based and recomputing with encoded operands approaches, we present the following schemes to detect both transient and permanent faults in all components of the cipher. The schemes are also shown in Fig. 1 and are described in details below.

A. Proposed Schemes for AddRoundKey

Our initial proposed scheme is applied to AddRoundKey (State, sk^i) which is the modulo-2 addition of the round-key sk^i, where $i\epsilon[1, N_R]$ and N_R is the number of rounds. In this scheme, the signatures of inputs are modulo-2 added to derive the output signature in each round, i.e., $\widehat{\text{Sig.}}(O) = \text{Sig.}(S_i) \oplus \text{Sig.}(sk^i)$, with "hat" denoting the predicted signature as shown in the bottom of Fig. 1 as a modulo-2 addition gate.

B. Proposed Schemes for RotateNibbles

The proposed signature-based error detection scheme is not confined to a particular signature. Furthermore, the signature-based scheme is applicable to the third operation in each round, RotateNibbles (State), which performs a left cyclic shift in inputs, e.g., parity signature of input nibbles are equal to the output nibbles signatures as shown in the left side of Fig. 1 in the RotateNibbles part.

C. Proposed Schemes for SubNibbles (S-Boxes)

The other operation in data processing, SubNibbles (state), is performed by substituting the state through 4×4 involutive S-boxes. For the KLEIN S-boxes, we propose a fine-tuned signature-based scheme taking into account two hardware implementation approaches, i.e., look-up table (LUT)-based and logic gate-based. Our case study of (interleaved) parity-based scheme is based on deriving the predicted parity bits to compare with the actual ones to detect errors. We propose the (interleaved) parity-based scheme for LUT-based approach of the 4-bit involutive S-boxes as shown in Table I.

In the second approach, for each 4-bit involutive S-box, we suppose the 4-bit input to the S-box as (a, b, c, d) and the 4-bit output as (a', b', c', d'), with $\vee$ representing an OR gate, then, we have the following equations:

$$\begin{aligned} a' &= \bar{a}\bar{c} \vee \bar{b}\bar{c}d \vee \bar{a}bd \vee abc\bar{d} \\ b' &= \bar{b}\bar{c}\bar{d} \vee \bar{a}\bar{b}\bar{c} \vee bc\bar{d} \vee db\bar{c} \vee a\bar{b}cd \\ c' &= \bar{a}\bar{b}\bar{d} \vee a\bar{b}c \vee acd \vee b\bar{c}d \\ d' &= a\bar{c}\bar{d} \vee ab\bar{c} \vee \bar{b}c\bar{d} \vee a\bar{b}c \vee \bar{a}\bar{b}\bar{c}d \vee \bar{a}bcd. \end{aligned} \tag{1}$$

Below are the predicted (interleaved) parities

$$\begin{aligned} \hat{P}_S &= \bar{d}\bar{c}a \vee \bar{a}cd \vee abc \\ \hat{P}_S^{(0)} &= c\bar{b}\bar{d} \vee \bar{a}\bar{c}b\bar{d} \vee ac\bar{d} \vee a\bar{b}c \vee a\bar{b}\bar{c}d \vee ab\bar{c}d \vee \bar{a}bcd \\ \hat{P}_S^{(1)} &= d\bar{b} \vee ad \vee a\bar{b}\bar{c} \vee \bar{a}b\bar{d}\bar{c}. \end{aligned} \tag{2}$$

D. Proposed Schemes for MixNibbles

In this section, the signature-based error detection schemes for the permutation component of KLEIN, MixNibbles, are proposed (also refer to Fig. 1). The operation inputs $\{a^i_0, a^i_1, \ldots, a^i_{15}\}$ in the ith round, are divided to two 8-nibbles in the form of polynomials over

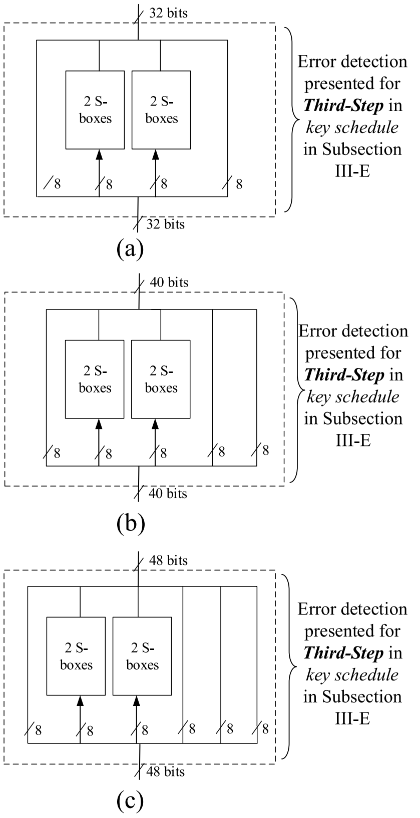
32 bits
2 S-boxes
2 S-boxes
8
8
8
8
32 bits
Error detection presented for ***Third-Step*** in *key schedule* in Subsection III-E
(a)
40 bits
2 S-boxes
2 S-boxes
8
8
8
8
8
40 bits
Error detection presented for ***Third-Step*** in *key schedule* in Subsection III-E
(b)
48 bits
2 S-boxes
2 S-boxes
8
8
8
8
8
8
48 bits
Error detection presented for ***Third-Step*** in *key schedule* in Subsection III-E
(c)

Fig. 2. Proposed error detection of key schedule for KLEIN. (a) KLEIN-64. (b) KLEIN-80. (c) KLEIN-96.

GF(2^4) in which each of these tuples is multiplied modulo $x^4 + 1$ with the following MDS matrix M to produce $\{s_0^i, s_1^i, \ldots, s_{15}^i\}$:

$$S = M \times C \Longrightarrow \begin{pmatrix} s_{0+j} \| s_{1+j} \\ s_{2+j} \| s_{3+j} \\ s_{4+j} \| s_{5+j} \\ s_{6+j} \| s_{7+j} \end{pmatrix}$$

$$= \begin{pmatrix} m_0 & m_1 & m_2 & m_3 \\ m_4 & m_5 & m_6 & m_7 \\ m_8 & m_9 & m_{10} & m_{11} \\ m_{12} & m_{13} & m_{14} & m_{15} \end{pmatrix} \times \begin{pmatrix} a_{0+j} \| a_{1+j} \\ a_{2+j} \| a_{3+j} \\ a_{4+j} \| a_{5+j} \\ a_{6+j} \| a_{7+j} \end{pmatrix}$$

$$M = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}. \tag{3}$$

We propose *column signature-based scheme* for respective error detection in which the modulo-2 addition of the each column in the output matrix S is equal to that of the input matrix A as shown in the left side of Fig. 1. The modulo-2 addition of each column of matrix M is equal to "1." The other signature-based scheme (interleaved cumulative column signature) is derived from modulo-2 addition of odd row elements with each other and likewise for the even ones, for example

$$s_0 \oplus s_4 = 3.a_0 + 2.a_2 + 3.a_4 + 2.a_6. \tag{4}$$

E. Proposed Schemes for Key Schedule

Different lengths of key (64, 80, 96) are applied to the KLEIN key schedule, in parallel with the data process unit. Since the required RoundKey for the cipher should be of the same-size as the 64-bit input and output, both of key schedule units for the key sizes of 80 and 96 bits provide the RoundKey output as the 64-bit subkey by considering just the leftmost 64 bits of the generated RoundKey. Therefore, our proposed error detection schemes for the key schedule of KLEIN are applicable to all of these key sizes (64, 80, and 96 bits) shown in Fig. 2, and we take the following steps for error detection.

Step 1: Each of the tuples has a one-byte left cyclic shift; thus, we propose the signature-based schemes similar to RotateNibbles.

Step 2: The Feistel-like structure of the key generation gives us freedom to propose two error detection schemes, i.e., signature-based and recomputing approach. In the former one, the signature of both tuple inputs before Feistel structure is equal to the output tuples signature. Given the potential limitation of the signature-based scheme in detecting all types of faults, using the proposed recomputations, one can expand the error detection surface.

Step 3: Each tuple can apply the recomputing with rotated operands approach (a variant of time redundancy scheme with the difference of rotating the tuples in the second round to detect the faults), and the signature-based error detection scheme. For instance, in the left tuple (most significant bits), the signature-based scheme is used for modulo-2 addition of the third byte with the counter number similar to AddRoundKey operation. This critical place is a key point for differential fault attack (DFA) for KLEIN to inject faults by changing the counter number i to generate a different key for the same round [4]. The right tuple which utilizes same S-boxes (one used in data process) in the third and second bytes, uses the same signature-based scheme for SubNibbles.

Our proposed schemes are capable of deteriorating classic and biased-fault attacks. For DFA attacks, in general, compared to the error detection schemes that are not practical for lightweight block ciphers, our proposed schemes get to high error coverage with low overhead (that can be customized). We have considered stuck-at fault models following the models in [11]. Applying the proposed schemes, covering both transient and permanent faults, provides efficient error detection in particular for the SubNibbles and MixNibbles operations. Nevertheless, the presented variants (parity and interleaved parity) are among other examples of the general signature-based schemes in LUT-based and logic gate structures; the merit of the proposed approaches is that they can be customized for all MDS matrices used in the lightweight SPN architectures.

TABLE II
THREE FAULT MODELS USED TO INJECT FAULTS FOR KLEIN-64 AND THEIR RESPECTIVE ERROR COVERAGE

Fault model	Injected faults	Fault detection
Two-bit	10,000	9,987 (99.87%)
	100,000	99,995 (99.99%)
Three-bit	10,000	9,932 (99.32%)
	100,000	99,974 (99.97%)
Multiple	10,000	9,957 (99.57%)
	100,000	99,984 (99.98%)

IV. ERROR SIMULATIONS AND FPGA BENCHMARK

Our schemes provide acceptable coverage with low-cost implementations as discussed in this section.

A. Error Coverage

The injected internal faults are modeled as transient and permanent faults which are considered throughout this paper. These single and multiple stuck-at faults occur in the ciphertext or intermediate states for which the single-bit errors are not simulated due to their

TABLE III
IMPLEMENTATION RESULTS FOR THE ORIGINAL KLEIN-64/80/96 ENCRYPTION AND OUR PROPOSED ERROR DETECTION SCHEMES ON VIRTEX-7 (XC7VX330T)

Architecture	Area (occupied slices)	Delay (ns)	Power (mW)	Throughput (Gbps)	Efficiency (Mbps/slices)
KLEIN-64 (logic gate-based)	105	2.53	200	25.28	240.76
Signature-based scheme	112 (6.66%)	2.79 (10.27%)	220 (10%)	22.93 (9.29%)	204.73 (14.96%)
KLEIN-80 (logic gate-based)	106	2.58	200	24.80	234.02
Signature-based scheme	113 (6.60%)	2.82 (9.30%)	223 (11.5%)	22.69 (8.48%)	200.79 (14.19%)
KLEIN-96 (logic gate-based)	108	2.63	209	24.33	225.27
Signature-based scheme	115 (6.48%)	2.94 (11.78%)	231 (10.5%)	21.76 (10.52%)	189.21 (16.00%)

TABLE IV
IMPLEMENTATION RESULTS FOR THE KLEIN-64 ENCRYPTION WITH OUR PROPOSED ERROR DETECTION SCHEMES ON TWO OTHER XILINX FPGA FAMILIES FOR SUBNIBBLES AND MIXNIBBLES

Family (Device)	Area (occupied slices)	Delay (ns)	Power (mW)	Throughput (Gbps)	Efficiency (Mbps/slices)
Spartan-6Q (xq6slx75)	94 (1.05%)	4.3 (0.70%)	184 (30.01%)	14.88 (0.72%)	158.29 (2.16%)
Zynq (xq7z045)	116 (8.41%)	2.58 (3.61%)	267 (13.61%)	24.80 (3.51%)	213.84 (11.23%)

perfect detection (100% error detection) by our proposed signature-based scheme. One merit of the error injection in this section is that it is general and can be applied to similar architectures. The proposed schemes would be applicable to the Advanced Encryption Standard (AES)-like ciphers regardless of hardware platforms. Noting that in practice, flipping the exact bit or byte might be impractical for attackers, we simulate two, three, and multiple stuck-at faults as well to verify the error coverage of our proposed error detection schemes as presented in Table II. Through using two fault injection experiments, i.e., 10 000 and 100 000 faults which are different in fault injection locations, type of faults, and their counts for KLEIN-64, the error indication flags are derived. A linear-feedback shift register-based architecture injects the faults in the test bench, applied to both parts of the KLEIN cipher. The simulation results show that the proposed signature-based schemes for both parts are capable of detecting stuck-at faults with high error coverage as shown in Table II. This aforementioned error coverage includes that of the control unit as well (we also note that we have injected the faults in the eventual comparison XOR gate; nevertheless, this very gate can be hardened with low overhead). Finally, we note that the example subsets of the proposed approaches include simple parity capable of detecting odd faults (including single faults which is the ideal case for the attackers), interleaved parity covering hardware defects resulting from burst faults, e.g., adjacent faults, and linear codes with the capability of detecting random errors of different multiplicity.

B. Implementation Results

We present the overhead assessment through FPGA implementation as shown in Table III. The Virtex-7 (xc7vx330t) FPGA family with VHDL as design entry in the integral of squared error version 14.7 has been utilized for the original and fault diagnosis structures in S-boxes (logic gate-based) and MixNibbles which are the most area consuming units. In general, all of the KLEIN types have the area and delay overheads under 7% and 12%, respectively, which shows the low cost of the proposed schemes. In order to benchmark the results on other FPGA platforms, we also present Table IV. This table summarizes the overheads of our proposed schemes (in parentheses) with respect to the original KLEIN encryption operation through two other Xilinx families (Spartan-6Q and Zynq) that are widely used. As a result, the proposed implemented schemes with the mentioned reasonable overheads and for high error coverage are efficient to realize more reliable hardware implementations of KLEIN. We note that although there has not been prior work on error detection of KLEIN, the overhead of the proposed error detection schemes is close to the fault diagnosis overheads of new lightweight cipher Midori (AES-like cipher) [12].

We note that, typically, in lightweight block ciphers, the most expensive components in terms of area are the S-boxes (SubNibbles) and the MDS matrices, which provide diffusion and confusion by nonlinear functions through S-boxes and linear transformations (XOR operation), respectively. This point has given us an accurate view to analyze the results of Table III, in which the implementation data for the proposed schemes for KLIEN-64 and KLEIN-80 are close or the same, with a minor variation for KLEIN-96. The first reason for such observation is that the difference of these types of KLEIN is primarily in the key schedule process which is minor, i.e., the number of modulo-2 addition gates with the same number of S-boxes. Moreover, the FPGA architectures do not closely follow the theoretical results due to mapping and place and routing processes.

C. Fault Space Transformation for KLEIN

In what follows, we also present the complications of adopting FST for lightweight block ciphers similar to KLEIN cipher. In FST for the AES [13], noting that biased fault attacks weaken redundancy-based countermeasures using the precise quantification of the bias of a fault model regarding the variance of the fault probability distribution, a generic mapping is done for data storage during encryption/decryption operations of AES-like ciphers. This increases the security of expensive redundancy-based countermeasures against both DFA- and DFIA-based attacks [13]. In other words, this approach changes the odds of occurrence of faults injected during the round operations by mapping them to different numbers with a special function.

We have applied this method to KLEIN. Implementing the "naive" spatial redundancy of KLEIN, we need 232 occupied slices on Virtex-7 and that shows the high area overheads with simple hardware spatial redundancy. Moreover, we have implemented the spatial redundancy with FST method that applies MixNibbles as a mapping function W through this redundancy and using InvMixNibbles as its inverse W^{-1}. We note that although KLEIN allows two types of decryption through: 1) encryption transformations but through modes of operations and 2) through reverse transformations, this might not be the case for other lightweight block ciphers and that adds to the complications of using FST. Our implementations show that higher area, i.e., 239 occupied slices, is achieved, as expected, due to the W function. Because of the KLEIN AES-like structure, applying the pipeline method, which utilizes MixNibbles operation as *W* function, improves the spatial redundancy algorithm with occupying 235 slices on Virtex-7.

V. Conclusion

In this paper, we have proposed efficient error detection schemes, i.e., signature-based and recomputing with encoded operands approaches, to increase the reliability of the cipher KLEIN. The proposed fault detection schemes are presented for different parts in key schedule and data processing including signature-based schemes for the only nonlinear component, S-box, with both logic gate-based and LUT structures. We have benchmarked the proposed architectures to assess their ability to detect transient and permanent faults by performing fault injection simulations. Moreover, we have implemented the proposed error detection architectures on different Xilinx FPGA families and adopted the FST approach for the KLEIN cipher. As a result, the proposed efficient error detection architectures can be feasibly utilized for KLEIN, are platform- and architecture-oblivious, and are suitable for the required performance, reliability, and implementation metrics of constrained applications.

References

[1] Z. Gong, S. Nikova, and Y. W. Law, "KLEIN: A new family of lightweight block ciphers," in *Proc. Radio Frequency Identification Security Privacy Issues*, Amherst, MA, USA, 2011, pp. 1–18.

[2] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Proc. Adv. Cryptol.*, Santa Barbara, CA, USA, 1997, pp. 513–525.

[3] D. Boneh, R. A. DeMillo, and R. J. Lipton, "On the importance of checking cryptographic protocols for faults," in *Proc. Int. Conf. Theory Appl. Cryptograph. Techn.*, Konstanz, Germany, 1997, pp. 37–51.

[4] H. Yoshikawa, M. Kaminaga, A. Shikoda, and T. Suzuki, "Round addition DFA on lightweight block ciphers with on-the-fly key schedule," *J. World Acad. Sci.*, vol. 17, no. 9, pp. 1743–1746, 2015.

[5] 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.

[6] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "A lightweight high-performance fault detection scheme for the advanced encryption standard using composite fields," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 1, pp. 85–91, Jan. 2011.

[7] 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.

[8] M. Mozaffari-Kermani, R. Azarderakhsh, and A. Aghaie, "Fault detection architectures for post-quantum cryptographic stateless hash-based secure signatures benchmarked on ASIC," *ACM Trans. Embedded Comput. Syst. Special Issue Embedded Device Forensics Security State Art Adv.*, vol. 16, no. 2, pp. 1–19, 2017.

[9] N. F. Ghalaty, B. Yuce, and P. Schaumont, "Analyzing the efficiency of biased-fault based attacks," *IEEE Embedded Syst. Lett.*, vol. 8, no. 2, pp. 33–36, Jun. 2016.

[10] N. Joshi, K. Wu, J. Sundararajan, and R. Karri, "Concurrent error detection for involutional functions with applications in fault-tolerant cryptographic hardware design," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 25, no. 6, pp. 1163–1169, Jun. 2006.

[11] T. Fuhr, E. Jaulmes, V. Lomné, and A. Thillard, "Fault attacks on AES with faulty ciphertexts only," in *Proc. Conf. Fault Diagnosis Tolerance Cryptograp.*, Santa Barbara, CA, USA, 2013, pp. 108–118.

[12] A. Aghaie, M. M. Kermani, and R. Azarderakhsh, "Fault diagnosis schemes for low-energy block cipher Midori benchmarked on FPGA," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 4, pp. 1528–1536, Apr. 2017.

[13] S. Patranabis, A. Chakraborty, D. Mukhopadhyay, and P. P. Chakrabarti, "Fault space transformation: A generic approach to counter differential fault analysis and differential fault intensity analysis on AES-like block ciphers," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 5, pp. 1092–1102, May 2017.