12-2016

# Lightweight Architectures for Reliable and Fault Detection Simon and Speck Cryptographic Algorithms on FPGA

Prashant Ahir
pa8238@rit.edu

# LIGHTWEIGHT ARCHITECTURES FOR RELIABLE AND FAULT DETECTION SIMON AND SPECK CRYPTOGRAPHIC ALGORITHMS ON FPGA

by

Prashant Ahir

A Gradute Paper Submitted

in

Partial Fulfillment

of the

Requirements for the Degree of

MASTER OF SCIENCE

in

Electrical Engineering

Approved by:

PROF. _____

(GRADUATE PAPER ADVISOR - DR. MEHRAN MOZAFFARI-KERMANI)

PROF. _____

(DEPARTMENT HEAD - DR. SOHAIL A. DIANAT)

DEPARTMENT OF ELECTRICAL AND MICROELECTRONIC ENGINEERING

COLLEGE OF ENGINEERING

ROCHESTER INSTITUTE OF TECHNOLOGY

ROCHESTER, NEW YORK

DECEMBER, 2016

I would like to dedicate this work to my family and friends for their love and support during my work on the project.

# Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other University. This dissertation is the result of my own work and includes nothing which is the outcome of work done in collaboration, except where specifically indicated in the text.

Prashant Chhabildas Ahir

December, 2016

# Acknowledgements

I would like to thank my advisor Dr. Mehran Mozaffari-Kermani for his guidance and feedback which helped in the successful completion of my graduate research. I also thank Dr. Reza Azarderakhsh for his constructive comments and participation.

# Abstract

The widespread use of sensitive and constrained applications necessitates lightweight (low-power and low-area) algorithms developed for constrained nano-devices. However, nearly all of such algorithms are optimized for platform-based performance and may not be useful for diverse and flexible applications. The National Security Agency (NSA) has proposed two relatively-recent families of lightweight ciphers, i.e., Simon and Speck, designed as efficient ciphers on both hardware and software platforms. This paper proposes concurrent error detection schemes to provide reliable architectures for these two families of lightweight block ciphers. The research work on analyzing the reliability of these algorithms and providing fault diagnosis approaches has not been undertaken to date to the best of our knowledge. The main aim of the proposed reliable architectures is to provide high error coverage while maintaining acceptable area and power consumption overheads. To achieve this, we propose a variant of recomputing with encoded operands. These low-complexity schemes are suited for lowresource applications such as sensitive, constrained implantable and wearable medical devices. We perform fault simulations for the proposed architectures by developing a fault model framework. The architectures are simulated and analyzed on recent field-programmable grate array (FPGA) platforms, and it is shown that the proposed schemes provide high error coverage. The proposed low-complexity concurrent error detection schemes are a step forward towards more reliable architectures for Simon and Speck algorithms in lightweight, secure applications.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Lightweight Cryptography

The need for lightweight cryptography has been emerged due to the advancement of constrained devices, such as radiofrequency identification (RFID) tags, nano-sensor networks, and applications such as implantable and wearable medical devices. These utilize sensitive, low-power implementations over very small chip area and consume low amount of energy. The Advanced Encryption Standard (AES), the current symmetric-key cryptography standard, may not achieve the necessary constraints for area, power consumption, and energy, necessitating use of lightweight block ciphers. There have been prominent efforts to make the AES more compact, e.g., a 128- bit AES was developed that expanded over an area of 2,400 gate equivalent [1]. This has been a considerable reduction in area considering the AES algorithm. However, it is still a large overhead burden for highly-constrained environments. Moreover, the AES cannot adapt to the varying level of security needed by different devices. Not all devices can spare area for 128-bit security. Consequently, it might waste chip area to encrypt 128-bit vectors where less bits need to be protected.

The above motivation calls for lightweight security and thus many lightweight block ciphers have been proposed to address these problems. However, some of these ciphers have

been optimized for high performance on either hardware or software platforms. The ciphers KATAN and KTANTAN [2], and PICCOLO [3] are all lightweight but are optimized to perform best on hardware platforms and might struggle to give good performance on software-based constrained devices. Similarly, for algorithms such as SEA [4] and LED [5] ciphers, having small code size and memory make them more inclined towards software-based devices, having a constrained instruction set.

Currently, ISO 29192-2 standard specifies two lightweight block ciphers: CLEFIA, 128-bit block cipher, and PRESENT, 64-bit block cipher. CLEFIA could provide high security along with good hardware and software implementation capabilities. It had a proven highest hardware gate efficiency of 401 on 90nm technology [3]. Moreover, it could perform on a wide range of processors at high speeds. Similarly, PRESENT [6] has a compact design smaller than the AES. It was optimized for hardware implementations by using a single 4-bit S-box and had low power consumption and high chip efficiency.

The National Security Agency (NSA) has proposed two new lightweight block ciphers - Simon and Speck [7], as alternatives to the above-mentioned encryption systems being used for RFID tag readers. These ciphers have been submitted to ISO for inclusion in ISO 29192-2 standard. They work better on small hardware devices which have memory and processor constraints. In [8], application-specific integrated circuit (ASIC) implementation of Simon and Speck was performed on 90nm technology and had efficiencies of 2,130 and 1,307, respectively. They use simple nonlinear functions like AND and modular additions which can be easily implemented on both hardware and software platforms, unlike PRESENT which has been optimized only for hardware implementations. Moreover, Simon and Speck are families of ciphers, and each family has different ciphers based on the sizes of the blocks and encryption keys. This makes them flexible to be used with a wide variety of devices. This is our motivation for choosing Simon and Speck families of block ciphers above the other lightweight block ciphers.

In [9] and [10], these ciphers have been analyzed by attacking some of the rounds, and it is concluded that the ciphers provide acceptable security. Differential fault analysis (DFA) of these ciphers has been carried out in [11]. The work has exploited the data leaking due to the AND operation in Simon to get the last round key. Similarly, in Speck, the modular addition has been proved to be the weak link giving out information to obtain the key. A proper fault detection technique needs to be in place to detect such cases and then respond to it by shutting down the device or deleting the secret key.

Concurrent error detection (CED) techniques have been widely used to architect reliable hardware for the AES and other cryptographic algorithms [12–19]. It is well-known that concurrent error detection techniques include a number of schemes, i.e., hardware, information, time, hybrid redundancy. Hardware redundancy makes use of extra hardware to process the same input twice to match the two outputs; any mismatch will trigger the error flag. Information redundancy schemes have a number of variants, e.g., parity codes [20] and robust codes [21]. Time redundancy technique has a number of schemes, i.e., recomputing with shifted operands (RESO) [22], [23], recomputing with rotated operands (RERO) [24], and recomputing with permuted operands (REPO) [25]. The hybrid redundancy scheme is given in [26–28] where different improvements in the architecture have been proposed. The choice of the CED technique is completely dependent on the requirements in terms of overhead tolerance, security, and reliability.

## 1.2 Fault Diagnosis

A fault in a system can be defined as a deviation from the expected working of the system which can be due to a defect of some components of the circuit. They can be temporary or permanent. Permanent faults are called as Solid or Hard faults and can result due to the wearing out or breaking of components. Temporary faults can be referred to as soft faults and these faults can be classified as intermiitent or transient as it occurs only at certain intervals of time.

An intermittent fault occurs when the component is developing a permanent fault. A transient fault can result due to some external disturbance like power supply fluctuations. Depending upon the effect of faults, they can be classified as parametric or logical. A parametric fault causes a change in speed, voltage or current as it alters the circuit parameter magnitude, while a logical fault ends up changing the Boolean function originally realized by the circuit. Delay fault which results due to slow gates is an important parametric fault and it leads to problems of critical races or Hazards. Fault extent can be local or distributed. A distributed fault affects multiple variables, whereas a local fault affects single variable. The clock malfunction is an example of a distributed fault while a logical fault is an example of a local fault. With the VLSI technology developing, the number of components on a single chip are increasing drastically thus also increasing the probability of fault occurrence. Thus, this is an important research area.

### 1.2.1 Faults and Degrdation

Depending on the behavior of the system, logical faults represent the behavior of the system modeled. Logical faults has three important classes:

**A) Stuck-at-faults:** A single stuck-at-fault happens when either one of the inputs or the output of the logic gate is fixed at either a logic 1 (stuck-at-1) or a logic 0 (stuck-at-0). They can be denoted by abbreviations as s-a-1 and s-a-0 respectively. This fault model is a good representation for types of defects such as open circuits and short circuits. The stuck-at model can also represent multiple faults which results when multiple signal lines are stuck at logic 0 or logic 1.

**B) Bridging faults:** Bridging faults occur when two or more than two signal lines are accidentally connected together. They can be classified as:

i) Input Bridging: This bridging fault results when a definite number of primary input lines are shorted.

ii) Feedback Bridging: This happens when there exists a short between an input and an output line. This fault causes the circuit to either oscillate or convert to a sequential circuit. It may occur between two or more signal lines or between the terminals of the transistor. In CMOS circuits, depending upon the bridging resistance and the physical location, faults end up manifesting as either stuck-open or stuck-at faults.

iii) Non-feedback Bridging: This category includes all the other remaining types of existing bridging faults apart from the above two types. If two lines happen to be physically close to each other, the probability of them getting bridged is higher. In a positive logic, bridging fault is assumed to behave as wired-AND with the dominant logic value being 0. In a negative logic, bridging fault is assumed to behave as wired-OR with the dominant value being 1.

**C) Delay Faults:** Due to the occurrences of the statistical variations in the manufacturing processes, the probability of appearance of smaller defects which causes partial short or open in a circuit, increases. Due to these defects, the circuit fails in meeting the timing specifications without altering the logic function of the circuit. The transition of the signal might get delayed from 1 to 0, or vice versa due to a small defect. This is called as delay fault. They are of two types:

i) Gate Delay Fault: It helps in modeling defects which causes the propagation delay of the faulty gate to exceed the worst case value specified. It can be used to model isolated defects but not distributed defects.

ii) Path Delay Fault: It can be used to model both isolated and distributed defects. This fault occurs when the propagation delay exceeds its specified limit along a circuit path.

**D) Transition and Intermittent Faults:** These can be classified as Temporary faults. Majority of the malfunctioning in the digital circuits results due to the temporary faults and these are also difficult to detect and isolate. Transient faults are the non-recurring temporary faults which occurs due to the fluctuations of the power supply or the circuit exposure to some external radiation like $\alpha$-particle radiation. As there is no physical damage to the hardware, these

faults cannot be repaired and thus are major souce of failures. Intermittent faults results due to poor designs, loose connections, or due to components which are partially defective. They happen due to the deteriorating or aging of the components, external environmental conditions like vibration, humidity, temperature etc. Intermittent faults is based on the protection of the system from the physical environment through cooling, filtering, shielding etc.

In digital systems, errors can happen through variouis causes including alpha particles from package decay, cosmic rays creating energetic neutrons and protons, and thermal neutrons. In advanced process technologies, errors can occur due to device shrinking, reduced power supply voltages, and higher operating frequencies which increase the probability of transient errors which can significantly affect reliability of computations. In addition, single event upsets and single event transients are generated due to cosmic rays which create energetic protons and neutrons, thermal neutrons, random noise, or signal integrity problems all resulting in device errors.

Degrdation in digital circuits can happen in many ways such as:

- *Time-Dependent Dielectric Breakdown* causes the leakage current affecting the transistor gates to increase, it results in short circuit.
- The phenomenon of *Electromigration* causes the metal ions to migrate thus leading to voids and holes in interconnect. These can cause open or short circuits which can cause faults.
- The *Hot-carrier effect (HCE)* can cause the threshold voltage in CMOS transistors to increase and also results in the degradtion of electron mobility.

### 1.2.2 Fault Detection Techniques

The process of determining whether the circuit contains a fault or not is called as fault detection [29]. As it is important to counteract such natural faults in order to achieve fault immunity

and reliability, error detection has been an important part of a number of hardware architectures in different domains, including various arithmetic unit sub-components [30]. In previous work, reliable architectures have been devised to counteract natural or malicious faults [31], e.g., cryptographic architectures immune to faults through concurrent error detection [12]. The different fault detection strategies can be classified as follows:

**A) Concurrent Error Detection:** It helps in detecting the faults in the circuit concurrently with the normal operation of the circuit by making use of additional logic. It results in an error if the resulting output is found different than the predicted output by the checker unit [32].The error coverage can be improved greatly using the methods of duplication or including parity check registers in the circuit. For improving the error coverage, the trade-off with area or latency, or throughput can be made. The errors can be also detected by running the circuit twice, once with the original operands and the second time using encoded operands such that different outputs are obtained. The checker will raise the error indication flaf incase of a mismatch between the two ouputs. The operands can be encoded using different methods like Recomputing with Shifted Operands (RESO), Recomputing with Rotated Operands(RERO), also by a slight modification of the RESO model [33].

**B) Off-Line Fault Detection:** This method helps in identifying faults in FPGAs and ASICs when they are not in operation with the use of additional circuitry. It helps in detecting manufacturing defects. Automated-Test-Pattern-Generator (ATPG) and Built-in-Self-Test (BIST) are some examples of off-line test circuits. The fault detection process does not involve the original circuitry. It connects the device under test between a pattern generator and an output response analyzer. In order to obtain full error coverage, it is important to check the logic and interconnects and the configuration network. For the FPGAs [34], the need of a large number of test configurations has been eliminated as the additional testing circuitry is built into the development boards by most of the recent consumer grade FPGAs [35]. BIST does not interfere with the normal FPGA operation, and also covers clock networks and PLLs

which are complicated systems.

**C) Roving Fault Detection:** This method helps in pointing out the faulty location in the FPGA circuit. It checks for defects in the FPGA by scanning it entirely and replaces those defects with a test function. It basically helps in adapting the BIST techniques with minimum increase in the area. In the roving detection, the entire FPGA is split equally into a number of regions where one region carries out the BIST testing while the others undergo normal operations. The speed of the roving method depends on the speed of the roving cycle as well as on the operation time. It has been reported that the latency of the best roving methods is less than one second.

## 1.3 Objectives

In this thesis, motivated by the lightweight constructions of Simon and Speck, we propose CED schemes which have acceptable area and power overheads instead of being a burden for such constructions. To the best of our knowledge, research on developing reliable architectures for Simon and Speck have not been reported to date.

Our contributions in this thesis are summarized as follows::

- We use time redundancy concurrent error detection techniques and propose reliable hardware architectures for both Simon and Speck block ciphers. These schemes add acceptable overhead to the original designs, maintaining the lightweight property of the crypto-architectures.

- The proposed architectures are benchmarked for the ability to detect transient and permanent faults by performing fault injection simulations. The results of our error simulations show high error coverage for both of these block ciphers. The proposed fault detection schemes give error coverage of 100% and 99.98% for Simon and Speck, respectively, for a multi-bit random fault injection.

- Finally, we implement the architectures on FPGA platform (two FPGA families from Xilinx, i.e., Virtex-7 and Zynq-7000) to compare the performance and implementation metrics with the original Simon and Speck designs. The results show that the proposed designs have acceptable overheads with very high error coverage. The area, delay, and throughput overheads are acceptable for these two ciphers. For instance, for Speck and Virtex-7, the power overhead is negligible, and the area, delay, and throughput overheads are 11%, 3%, and 3%, respectively.

## 1.4 Thesis Outline

The structure of the thesis is as follows:

- CHAPTER 2: This chapter provides the preliminary information for the understanding of basic SIMON and SPECK operation.
- CHAPTER 3: This chapter is used as a motivating section to give details regarding various CED techniques and their shortfalls. Moreover, it presents the proposed design for reliable architectures..
- CHAPTER 4: In this chapter, the fault injection simulations are performed to determine the error detection capabilities of the proposed architectures. The proposed designs are implemented on FPGA and benchmarked.
- CHAPTER 5: This chapter discusses the scope for future work and provides conclusions.

# Chapter 2

# Preliminaries

This chapter presents a brief description of Simon and Speck in following.

## 2.1 Simon

The Simon family has block ciphers for ten distinct block and key sizes which are generally written as Simon $2n/mn$ for a $2n$-bit block and $m$-word ($mn$-bit) key. For example, if the block size is 48 bits, then, $n$ = 24. If the word size is $m = 4$, then, key is $m : n = 4 : 24 = 96$ bits, i.e., $mn$ bits. The different sizes make the algorithm useful for a wide variety of constrained devices with different levels of security.

The round function is repeated to obtain a cipher-text and is a Feistel Map having two stages (see Fig. 2.1) as follows: $R_k(x;y) = (y \oplus f(x) \oplus k; x)$, where $f(x) = (S_x : S_x^8)S_x^2$, and $k$ is the round key given by the key schedule. In this process, $\oplus$denotes XOR, and for a given $j$, $Sj$ is left circular shift (the non-linearity is achieved here by rotating the same input by different number of bits and then performing their AND operation).

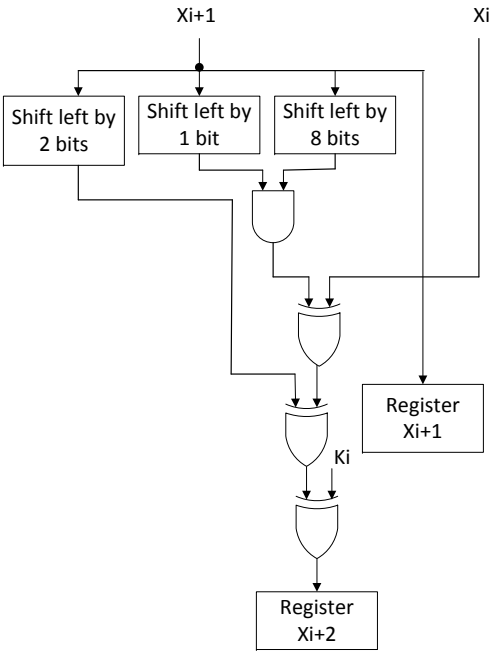


Figure 2.1: SIMON block diagram

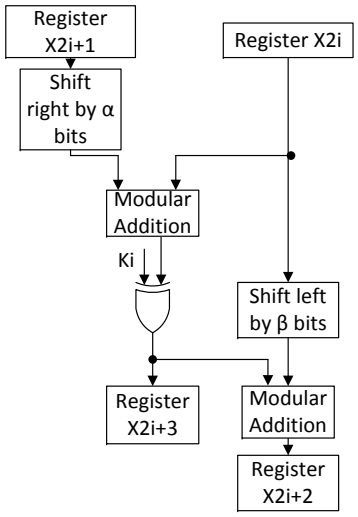


Figure 2.2: SPECK block diagram

## 2.2 Speck

The Speck family is represented, similar to Simon, as Speck 2n/mn. The round function, as shown in Fig. 2.2, is $R_k(x,y) = ((S_x^{-\alpha} + y) \oplus k, S_y^{\beta} \oplus (S_x^{-\alpha} + y) \oplus k)$. Here, if the block size is 32, then, inputs are rotated by amounts $\alpha = 7$ and $\beta = 2$ (similarly, $\alpha = 8$ and $\beta = 3$ for others). The non-linearity is obtained by using the modular addition which favors a software platform over hardware.

There is always a bargain between efficiency and security depending upon the application requirements. It is very difficult to achieve both at the same time. To obtain high level of security, a very strong algorithm with large key is needed but this increases the hardware overhead. Conversely, if efficiency is important, then we use a simple algorithm with a small key and run large number of rounds. This would not have large hardware overhead but the security obtained would not be very high. The Simon and Speck families with different key sizes for different block sizes attempt to give fairly good security, keeping hardware overhead to a low amount, nonetheless, giving good efficiency.

# Chapter 3

# Proposed Reliable Architectures for Simon and Speck

We present the motivation behind our work in this chapter and discuss shortfalls and problems encountered by different CED techniques. Then, we present our proposed CED technique for Simon and Speck.

## 3.1 Motivations

As motivations to our proposed work, we briefly present different CED techniques, and some possible shortcomings with respect to lightweight applications. Full hardware redundancy techniques give good fault detection architectures; however, this is at the cost of large hardware resources. Therefore, such schemes cannot be used for lightweight algorithms.

### 3.1.1 Signature-based Diagnosis Approach

The registers in the datapath are key elements to propagate the errors. Hence, it is imperative that we detect presence of faults in the datapath registers. Signatures, e.g., interleaved or single/multiple parity bits, can be efficiently used to represent the data held by the registers.

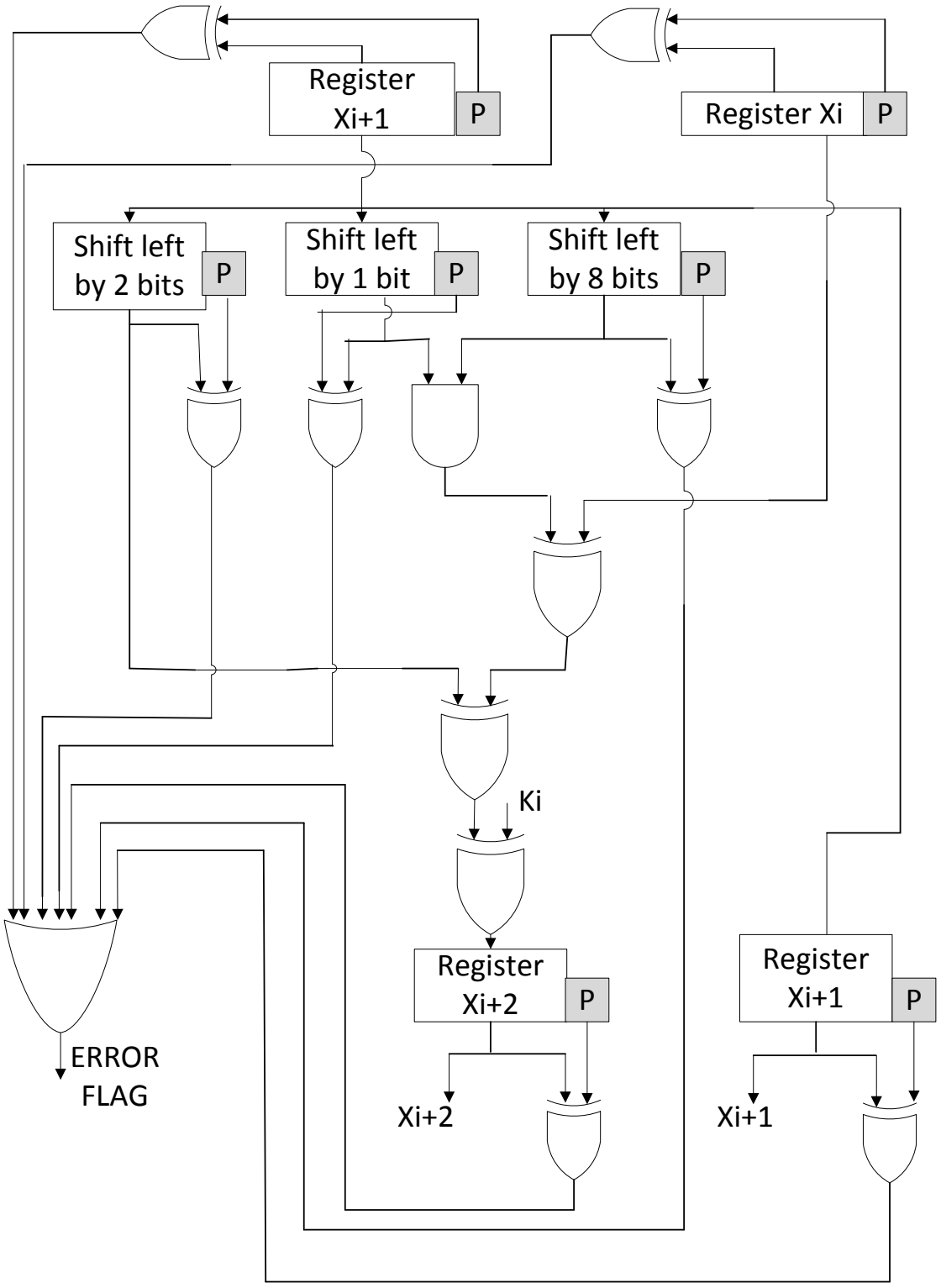


Figure 3.1: Proposed signature-based CED scheme for Simon.

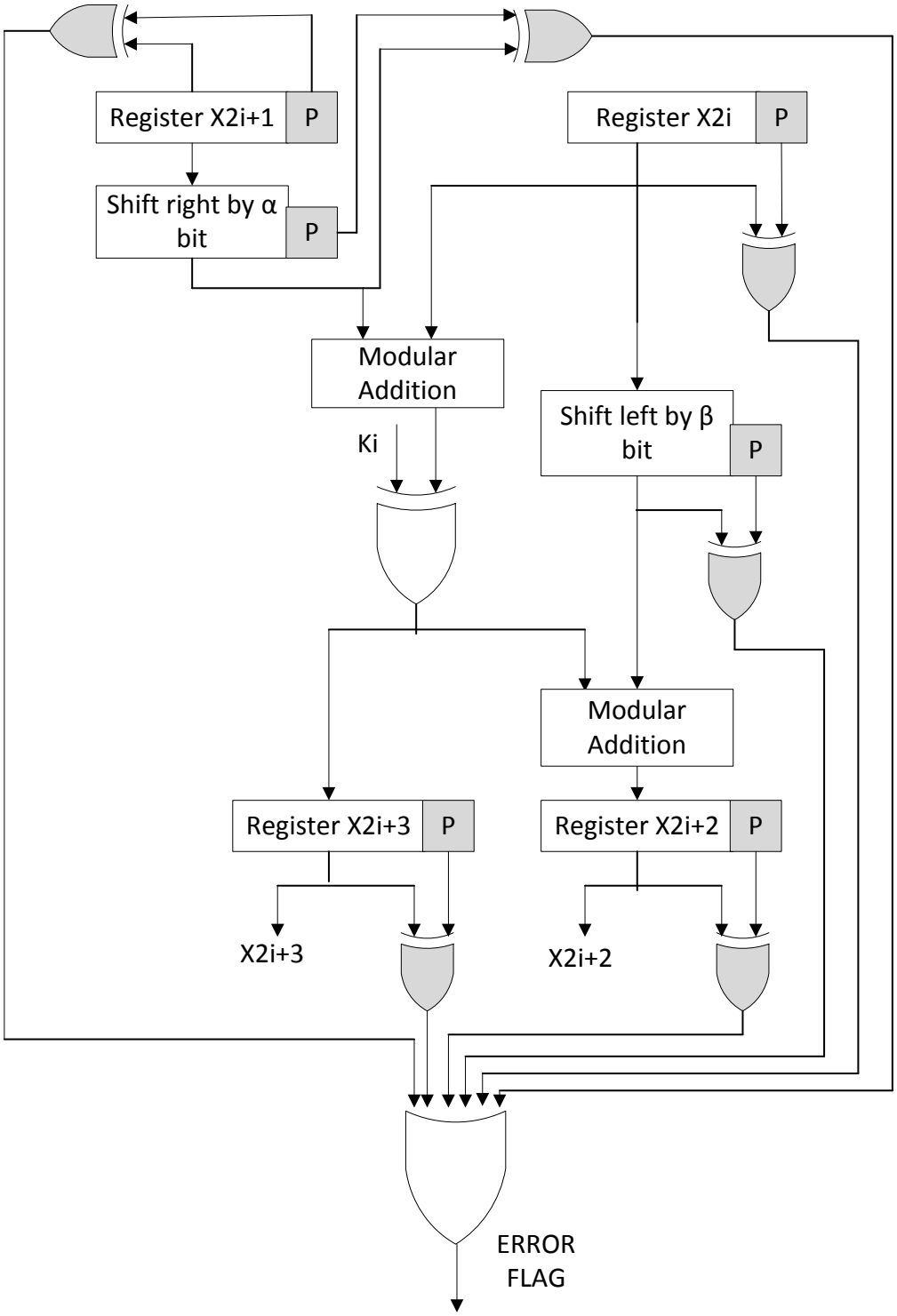


Figure 3.2: Proposed signature-based CED scheme for Speck.

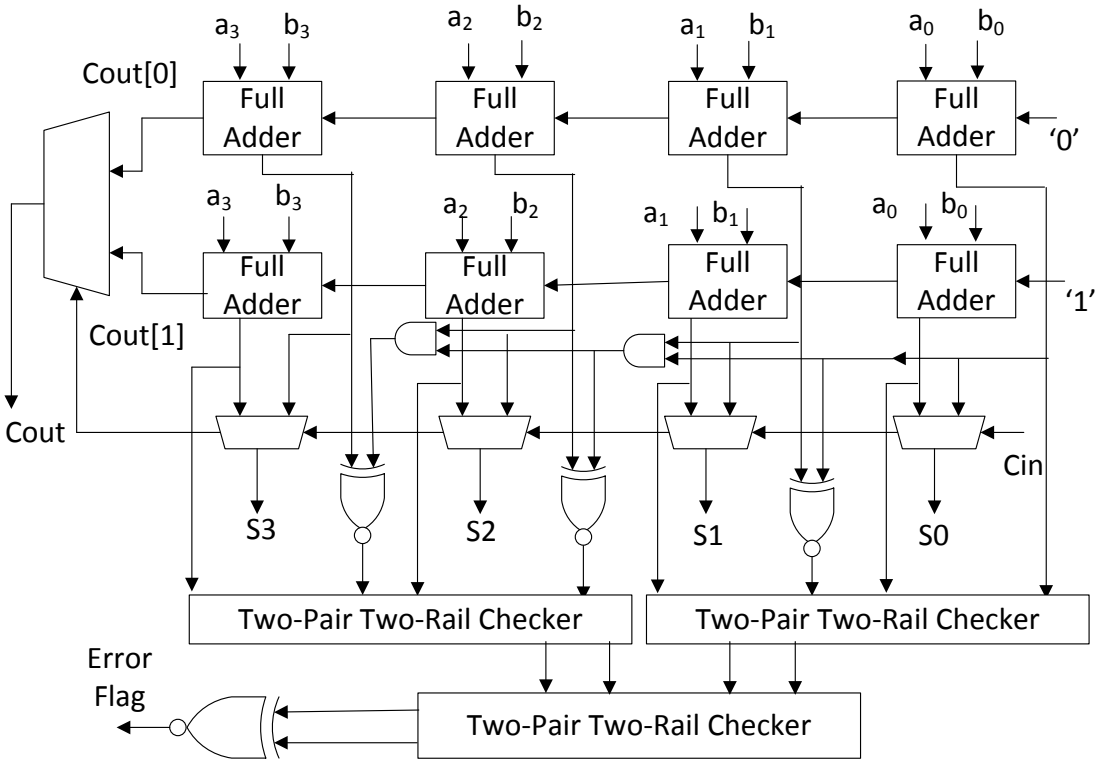


Figure 3.3: Adopted self-checking adder used for modular addition of Speck.

As a case study, parity-based CED scheme for Simon and Speck has been described in Fig. 3.1 and Fig. 3.2. The general approach is to calculate value for the parity bit based on the individual bits held by the register and then compare it by taking an XOR with the predicted parity bit value, then, any discrepancies witnessed raises the error indication flags.

The main disadvantage of the parity scheme is that the error coverage is almost only 50%. This is due to the fact that only odd number of faults can get detected with this method.

The Speck algorithm employs a modular adder in one of the steps to arbitrate the plaintext. To perform this addition, Fig. 3.3 shows a four-bit self checking adder. It uses two four-bit full adders to calculate addition results with input carry '0' and '1'. Then, according to actual input carry, the final output carry bit is selected. The self-checking action is performed by the two-pair two-rail checker as explained in [36]. However, as Speck and Simon are lightweight and are used in constrained applications, the aforementioned approach may not be suitable for

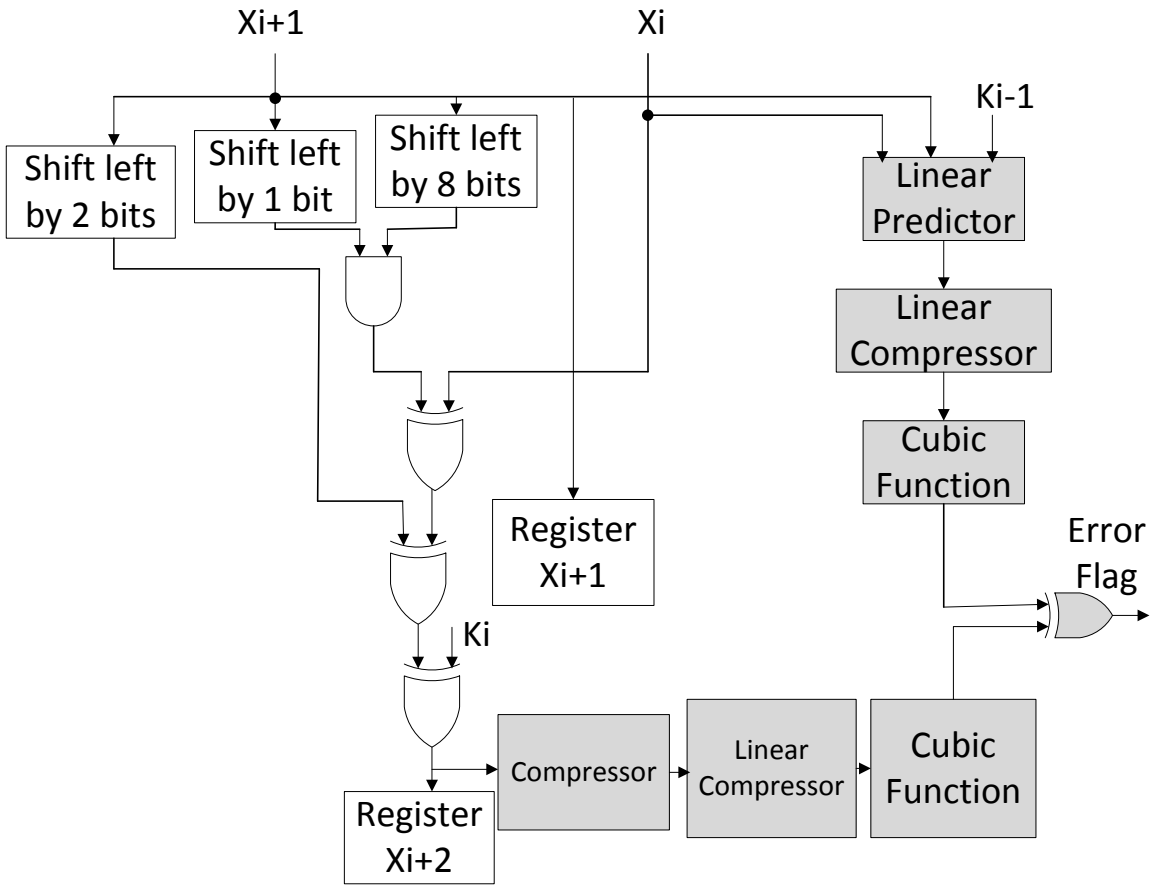


Figure 3.4: Applicable robust protection scheme for Simon.

error detection in our architecture.

### 3.1.2 Robust Protection Scheme

In [21], the authors have proposed a robust protection scheme against DFA attacks. It is based on using non-linear robust error detecting codes with input as well as the computed output. The proposed design employs a counter to count the number of faults encountered by the device in its life-time, and once it reaches a pre-decided threshold value, the secret key is cleared by the device since it is assumed that, typically, it encounters lower number of natural faults than those required by a practical DFA.

In this scheme, non-linear codes are obtained using a cubic function. As shown in Fig.

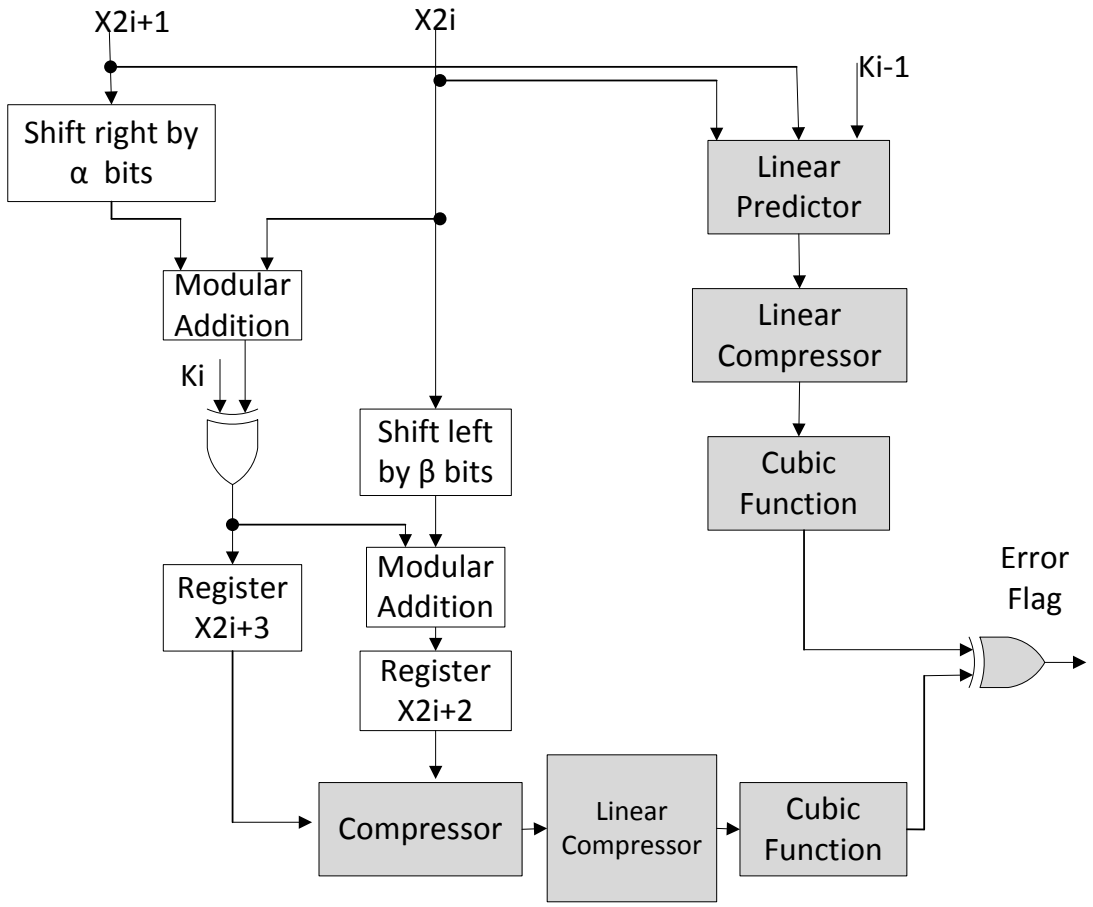


Figure 3.5: Robust error detection scheme for Speck.

3.4 and Fig. 3.5, applying such methods to Simon and Speck can be considered, i.e., two cubic functions are used at the input and output of each round function. The cubic function selection is based on the fact that it gives best error coverage without requiring complicated hardware. A square function does not give a good error coverage, and the functions having powers higher than three result in much complicated hardware. Thus, the cubic function is a trade-off between error coverage and complexity.

In these figures, the linear predictor generates a signature which is equivalent to the component-wise XOR of the output bytes of a round based on the block size. This signature is then passed on to the cubic function. In cases where the size of the cubic function is less than that of linear predictor, a compressor is used to compress the size of the predictor output so that it

matches the size of the cubic function input. The cubic function with signature $r$ is dependent on primitive polynomial. For a 24-bit input, the signature of the cubic function can be chosen to be less than or greater than 24. The compressor is needed if $r$ is greater than 24. In order to reduce the complexity by not using the compressor, the primitive polynomial can be, for instance, $x^{20}+x^{17}+1$ or $x^{16}+x^5+x^3+x^2+1$ for $r$ = 20 or 16, respectively. The compressor shown in the design is to illustrate a generalized architecture incorporating all components of a robust scheme. This scheme provides protection for the encryptor and decryptor, as well as the key generation algorithm.

This method gives 100% error coverage; however, the hardware overhead is almost 50% which may not be acceptable considering the lightweight applications of Simon and Speck. Thus, this scheme may not be ideal to be used for protection of Simon and Speck.

## 3.2 Proposed Error Detection Schemes

So far, we have explained problems with usage of various fault diagnosis schemes, such as higher overheads in case of hardware redundancy and robust codes or lower error detection rate in case of parity schemes. Therefore, we select a protection scheme that will provide close to 100% error coverage at suitable area and power overheads. The proposed scheme, as explained in the following, have high error detection rate at acceptable performance metric overheads.

Here, we propose concurrent error detection schemes which are applicable to both Simon and Speck.

In addition to the schemes used in this thesis, the RESO approach can also be used for error detection. In RESO-$k$, in the re-computation step, the inputs are shifted left by $k$ bits. Now, usually the leftmost $k$ bits, on shifting, will get lost. If we are to store them, we will need to house an $n+k$ bit register. This will, in turn, create needs for all the subsequent registers and computations to be of $n+k$ bit length, i.e., the adders and data-path registers will be of

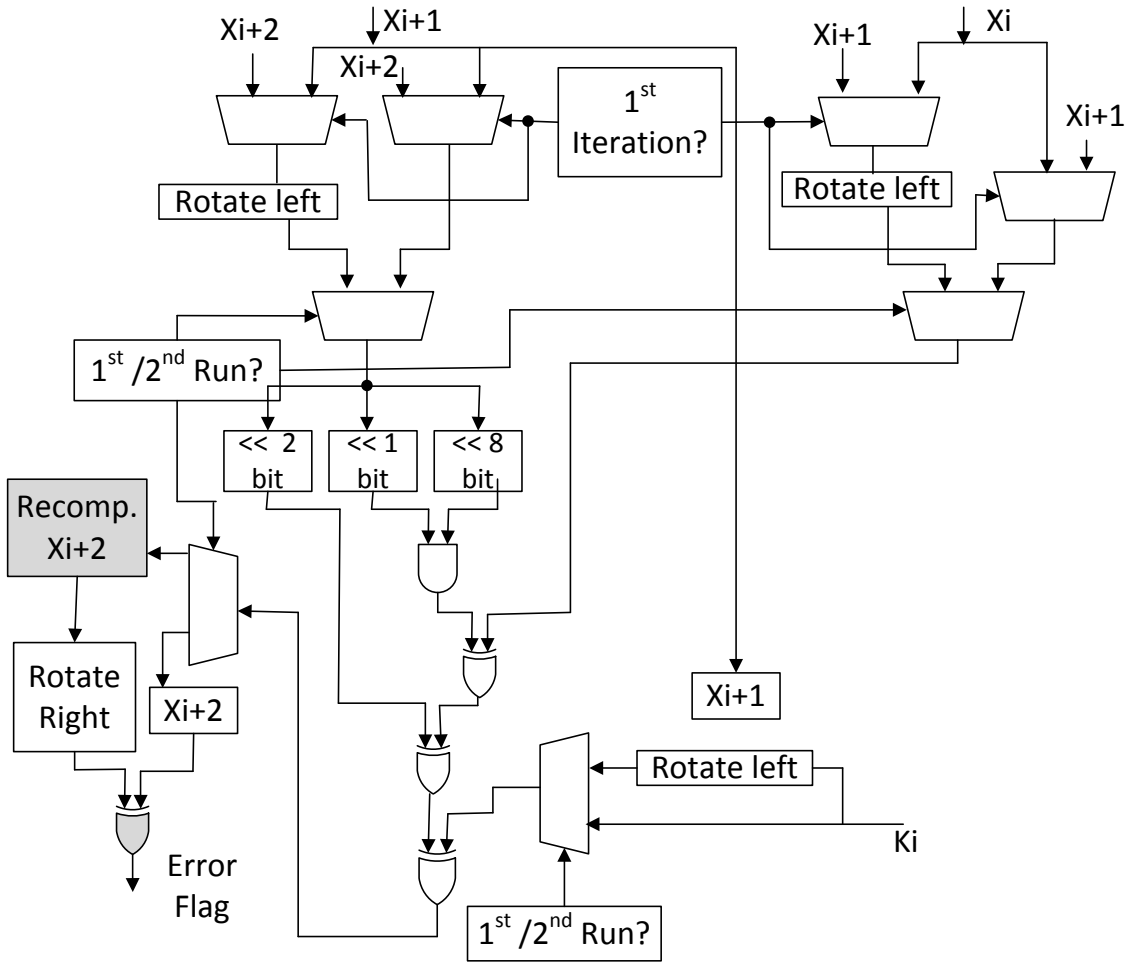


Figure 3.6: Proposed error detection approach for Simon.

$n+k$ bits. Due to this, the re-computation step will take more cycles to produce the output. This latency will only increase with higher values of $k$. To house the increased size, more chip area will be consumed, resulting in increased complexity. Due to these drawbacks, we do not propose RESO as a comparatively good error detection approach.

For the sake of brevity, we shall discuss only the error detection of the encryption operation. We note that the decryption can be protected through the proposed approaches as well.

We propose RERO for both Simon and Speck. Based on the methods of processing the data, we propose two types of architectures, i.e., iterative and pipelined architectures. A Si-

mon block cipher having a $2n$-bit block of plain-text made up of two $n$-bit words $Xi+1$ and $Xi$ is passed as input as shown in Fig. 3.6. Each of the input blocks (plaintext blocks) is operated upon twice. A multiplexer controls the passage of the normal and recomputed plaintext. During the first run, the operands are passed in their normal state. As can be seen in this figure, the Feistel stepping of Simon round function operates on the plaintext. The output generated is stored in a register for a later comparison. During the second run, the multiplexer selects the re-computed operand to be passed on to Simon. The recomputed plaintext is obtained by rotating the input by a constant value of $a$ bits. Each word of the input blockcipher is rotated by same amount of $a$ bits towards right or left. Similarly, the key $K_i$ is also recomputed by rotating it by same amount in same direction as the plaintext. The Feistel stepping function's output is the recomputed output. This output is then rotated in inverse direction by $a$ bits. The output thus obtained is compared with the output calculated originally in the first run. These two are then XOR-ed to check their equality and the error indication flag is raised if they are not equal.

Let us take the example of Simon48/96. Each of the 24-bit words of the input is transformed to $[Xi+1_{23}\ ..\ Xi+1_{j+1}\ Xi+1_j\ ..\ Xi+1_0]$ and $[Xi_{23}\ ..\ Xi_{j+1}\ Xi_j\ ..\ Xi_0]$. During the second run, we rotate left by $j$ bits ($j$ is an integer such that $j = 0$ to23) which makes the input as $[Xi+1_j\ ..\ Xi+1_0\ Xi+1_{23}\ ..\ Xi+1_{j+1}]$ and $[Xi_j\ ..\ Xi_0\ Xi_{23}\ ..\ Xi_{j+1}]$. The outputs, $Xi+2$ and $Xi+1$, are then fed back as inputs to the next round of the function. Thus, we *iterate* the input through the round function repeatedly to get a final secure ciphertext. A multiplexer selects between the main plaintext and the ciphertext generated by previous round. Each cipher family is iterated through a pre-decided number of times [7]. Simon48/96 is run through the round function 36 times to get the final output.

For the Speck algorithm, as shown in Fig. 3.7, a similar methodology is followed where we compare recomputed and original outputs. Inequalities will raise the error indication flags.

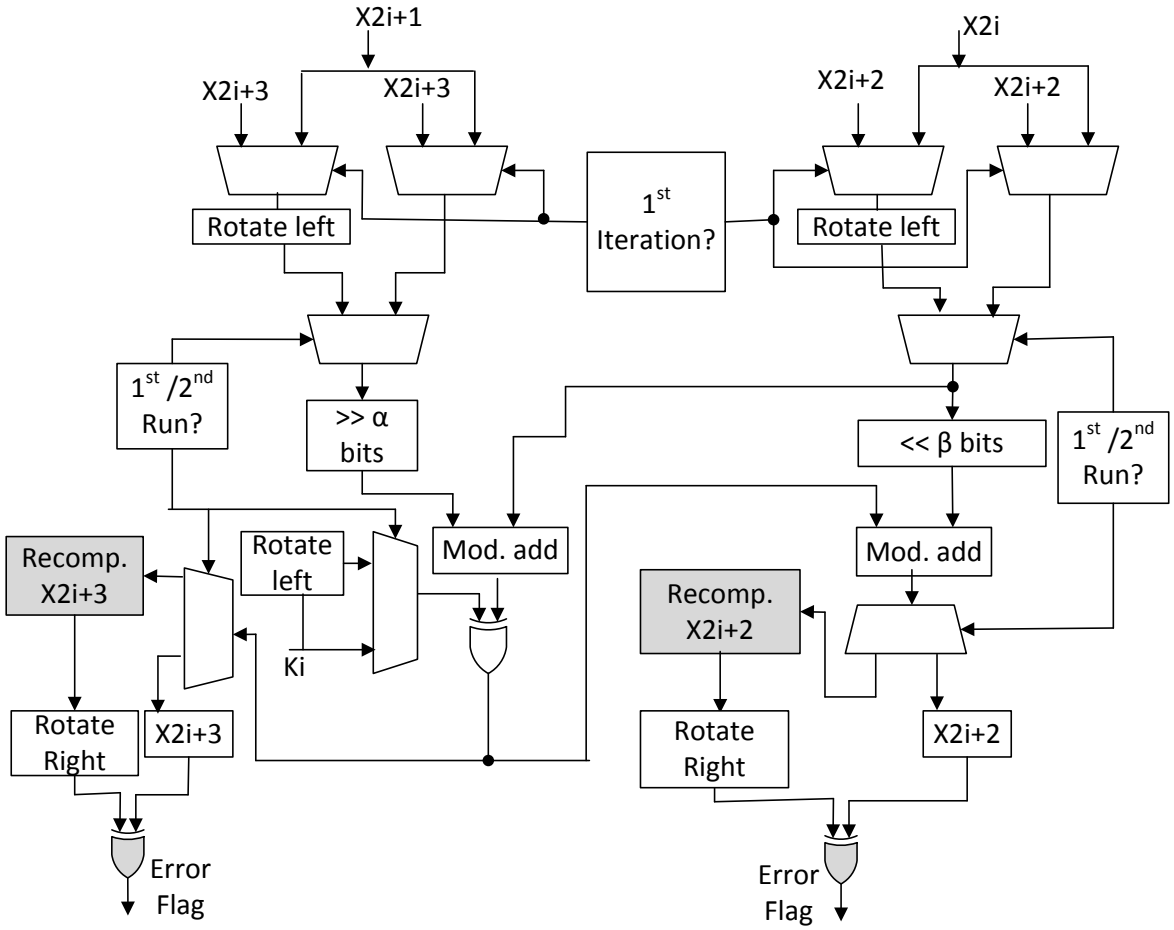


Figure 3.7: Proposed error detection approach for the Speck algorithm.

An important component in Speck is the modular adder. We use the self checking adder in Fig. 3.8 and modify it to make it work on rotated operands for Speck48/96.

Consider Fig. 3.8 which shows the proposed modified self checking adder. The main reason behind modifying the normal self-checking adder is to assure that the carry generated by addition of $b_{23}$ and $a_{23}$ does not affect the $z_0$ bit after rotation and after rotation, correct carry goes into addition of bits $b_{j+1}$ and $a_{j+1}$.

During the first run, the input operands are appended with bit-@ (a stuck-at-0 bit) at the most-significant bit position of both the operands. Therefore, an $(n+1)$-bit adder is needed to operate upon these operands. The effect of this bit-@ is such that no matter what carry out actually gets generated by the bit-$(n-1)$, the bit-@ will always be '0'. Now, during the

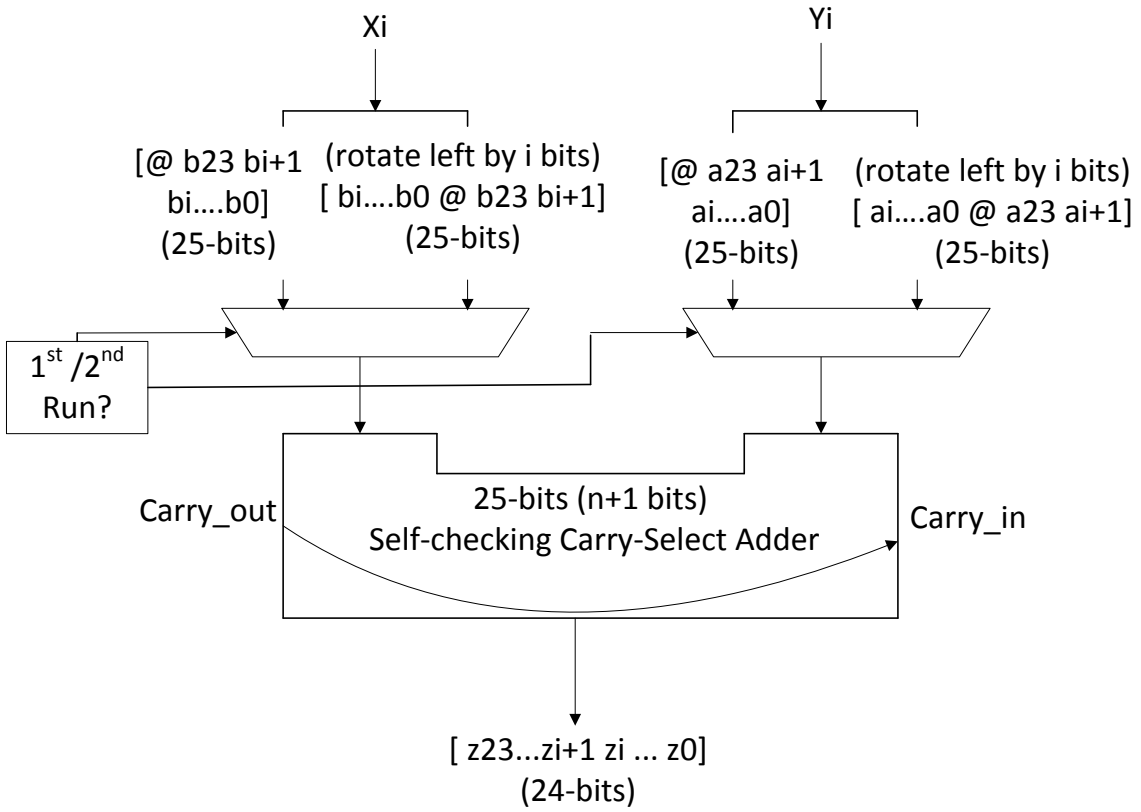


Figure 3.8: Modular adder operation for the Speck algorithm in the proposed scheme.

second run, after rotating the input, we ensure that the adder operands have bit-@ between bit-0 and bit-23. The addition of the bits-23 ($b_{23}$ and $a_{23}$) will generate a carry-out which does not affect the bits-0 ($b_0$ and $a_0$) addition result due to the presence of bit-@ between them. This enables correct addition result before and after the rotation. Moreover, as can be seen in the figure, the carry-out generated by the last bits is connected as carry-in to the first bits. This is again to ensure correct addition of the bits $j+1$ and $j$ during both runs. At the output $z$, the bit-@ is removed from the result, i.e., for first run, let $p$ be the output $[@\, b_{23} \ldots b_{j+1}\, b_j \ldots b_0]$ $+ [@\, a_{23} \ldots a_{j+1}\, a_j \ldots a_0] = [p_{23} \ldots p_{j+1}\, p_j \ldots p_0]$ and for the second run, let $q$ be the output $[b_j \ldots b_0 @\, b_{23} \ldots b_{j+1}] + [a_j \ldots a_0 @\, a_{23} \ldots a_{j+1}] = [q_{23} \ldots q_{j+1}\, q_j \ldots q_0]$.

In this iterative approach, we let the entire input pass through the hardware before passing the next input. This reduces the throughput since hardware is not being used at its fullest and it takes more number of cycles to run the input through single round. We, alternatively,

propose a pipeline architecture to improve such throughput degradations. Sub-pipelining can be performed to alleviate this problem. Suppose $n$ pipeline-registers have been placed to sub-pipeline the structures to break the timing path to approximately equal segments. Let us denote the segments of pipelined stages by $\exists_n$. The original input is first applied to the architecture and in the second cycle, while the second segment of the circuit executes the first input, the second input or the encoded variant of the first input is fed to the first half of the circuit (this depends on the objectives, i.e., reliability *vs.* getting the results first). This trend is consecutively executed for normal and rotated operands. Such approach ensures lower degradation in the throughput (and achieving higher frequencies) at the expense of more area overhead.

# Chapter 4

# Error Injection Simulations and Implementations through FPGA

In what follows, we present the results of our error simulations and FPGA implementations benchmark.

## 4.1 Error Simulations

The proposed fault detection architectures 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 that has been followed for the proposed fault diagnosis schemes is to inject faults and then observe the error indication flags. For simulations, Verilog HDL has been used. We have considered all the sub-blocks of the original architecture, i.e., the adders, XOR, AND, and OR gates, to induce faults by flipping one or more bits and then inspect the generated outputs. We have considered a particular fault scenario and applied different inputs to assert a sub-set of entries while injecting faults. We, then, observe all the errors that get detected for all the inputs. The fault model used to test the proposed architectures is created using external feedback Linear Feedback Shift Registers

(LFSR) to generate pseudo-random fault vectors that can flip random bits in the output of the gates and at random intervals. The LFSRs used here are 8-bit registers with the polynomial $x^8+1$ for maximum taps. This is achieved using multiplexers whose select signal is driven using an LFSR, thus, randomizing the selection of faulty bit (coming from another LFSR) and correct bit, i.e., the actual results.

As discussed in the previous chapters, for the RERO scheme, we pass normal input in the first round and then pass the rotated one in the second round. Thus, each of Simon and Speck requires two runs to detect presence of faults. The Simon block cipher has a combination of AND and XOR gates. We select random 4 bits from each of these gates and inject faults in them. The Speck block cipher, has two modular adders in its architecture as well. Similar approach is followed to induce faults here, i.e., we select any 4 bits from each of the adders and the gates and flip them using the LFSRs. Thus, a total of 12 different faults are induced in each Simon and Speck. In addition to this multiple random fault model and to assess other potential scenarios, we also test our architecture for 2/3/4-bit fault models. Overall 100,000 faults are injected in each cipher and the error indication flag is observed. A counter is set to count the number of faults detected. It is observed that for Simon we get very close to 100% error coverage, and for Speck, we get 99.98% error coverage, i.e., 99,980 faults are detected (for the sake of brevity, the tables for such faults are not added). This is inline with the expected results. Thus, our proposed architectures give very high fault coverage.

Next, we describe the results obtained after implementing our proposed architectures on Virtex-7 and Zynq-7000 Xilinx FPGA families [36].

## 4.2 Implementations on FPGAs

This part presents the overhead incurred while applying the proposed error detection schemes on FPGA platforms. We would like to emphasize that the presented results are independent of the platform or FPGA family, and similar results are expected on other hardware platforms.

The implementations on FPGAs have been performed on Xilinx Virtex-7 and Zynq-7000 families using Xilinx Vivado 2014.4 Design Suite. The devices used are xc7k70tfbg484-1Q and xc7v585tffg1157-3 from Zynq-7000 and Viretx-7 families, respectively. In order to get the overheads, we compare the implementation results obtained from the original Simon and Speck architecture with those from with the proposed error detection architectures.

The implementations have been performed for Simon48/96 and Speck48/96 block ciphers. The Simon cipher has to make 36 runs to give a final cipher text. Similarly, for Speck, it has to be run 23 times. According to the RERO approach, during each run, the input needs to be passed for two rounds in order to detect an error. This degrades the overall throughput of the device as can be seen in all the tables below. Nevertheless, we can alleviate this as discussed before using sub-pipelining.

Each of the two ciphers has a control unit that directs the passage of normal/rotated operands to the main block cipher module. The control unit then receives the normal/recomputed outputs at the end of each round and sets/resets the error indication flags.

The overhead calculations are made for three parameters, i.e., area overhead in terms of number of LUTs, delay (in terms of *ns*), and power consumption (in terms of *mW*), shown in Tables 4.1 to 4.4.

Table 4.1 and Table 4.2 outlay implementation results for Simon block cipher using Zynq-7000 and Virtex-7 families, respectively. The results are in conformity with our expectations for lightweight applications. As can be seen, the power and delay overheads for both devices are acceptable. The original Simon architecture being made up of combinational logic, has a small slice area occupancy. The XOR and OR gates, responsible for the setting of error indication flag, occupy considerable LUTs and hence the area overhead goes to roughly 30%. This can be seen as a trade-off for this scheme but considering that other viable error detection schemes consume more area, we can consider that this is an acceptable area overhead that is always incurred if the Simon block cipher is to be given close to 100% error coverage.

Table 4.1: Zynq-7000 FPGA implementation results for Simon block cipher

| Metric | Simon | Simon-RERO | Overhead |
|---|---|---|---|
| Power (mW) | 239 | ∼239 | Negligible |
| Delay (ns) | 5.448 | 5.607 | 2.919% |
| Area (LUT) | 73 | 95 | 30.137% |
| Throughput (Gbps) | 0.245 | (0.238)[1] | (2.836%) |

1. One stage sub-pipelined.

Table 4.2: Virtex-7 FPGA implementation results for Simon block cipher

| Metric | Simon | Simon-RERO | Overhead |
|---|---|---|---|
| Power (mW) | 248 | ∼248 | Negligible |
| Delay (ns) | 4.415 | 4.562 | 3.330% |
| Area (LUT) | 73 | 95 | 30.137% |
| Throughput (Gbps) | 0.302 | (0.292)[1] | (3.4%) |

1. One stage sub-pipelined.

Tables 4.3 and 4.4 provide the results for the Speck cipher implemented on Zynq-7000 and Virtex-7. It can be seen that for Speck, the metrics have low to acceptable overheads. Thus, our proposed error detection schemes give almost 100% error coverage at acceptable power, area, and delay overheads and, hence, can be used for the error detection of constrained lightweight Speck block cipher. The above proposed architectures are independent of the platforms considered. Even-though the implementations were performed only on select FPGA families, we expect similar results on other FPGA families and also on other hardware platforms.

Table 4.3: Xilinx Zynq-7000 FPGA implementation for Speck block cipher

| Metric | Speck | Speck-RERO | Overhead |
|---|---|---|---|
| Power (mW) | 234 | ∼234 | Negligible |
| Delay (ns) | 5.552 | 5.904 | 6.340% |
| Area (LUT) | 199 | 221 | 11.055% |
| Throughput (Gbps) | 0.376 | (0.353)[1] | (5.962%) |

1. One stage sub-pipelined.

Table 4.4: Xilinx Virtex-7 FPGA implementation for Speck block cipher

| Metric | Speck | Speck-RERO | Overhead |
|---|---|---|---|
| Power (mW) | 245 | ∼245 | Negligible |
| Delay (ns) | 4.183 | 4.333 | 3.585% |
| Area (LUT) | 199 | 221 | 11.055% |
| Throughput (Gbps) | 0.499 | (0.481)[1] | (3.6%) |

1. One stage sub-pipelined.

## 4.3 Differential Fault Analysis (DFA)

The proposed methods, being for reliability, can deal with permanent and transient faults. Even though the proposed methods make a potential DFA attack more difficult to mount but they may not completely thwart such attacks. Here, we present previous DFA attacks on Simon and Speck families and make additional modifications to our proposed architecture so as to go towards making such attacks more difficult.

The work in [37–39] presents three DFA attacks on Simon family. The authors used data leaked by AND operation to deduce the secret key. In case of Speck, the authors in [37]

describe that the modular addition can be used by the attackers to gain knowledge of the secret key. After analyzing the block cipher, they concluded that injecting fault in each round will not help them get the secret key. In [37] and [38], the authors have demonstrated that by injecting a bit-flip fault at the input of penultimate round (or ante penultimate round in case of [39]), they can deduce the value of at most two bits of the penultimate input. Thus, in turn, they can find out the value of the secret key used in the last round. The main difference in the three papers is the number of fault injections required to get all the bits of the secret key.

This DFA attack can potentially bypass the proposed RERO error detection scheme (please also refer to [40]). Therefore, we make a small architectural addition to our proposed scheme in order to detect such type of DFA attacks. Since the fault injections are made at the input of a round, we compare the input sub-cipher in each round (starting from second round) with that generated in previous round. Any discrepancies will be indicated by the error indication flag. Should the attacker try to inject faults in the sub-cipher in the previous round itself, the previously proposed RERO scheme will detect such an attack. Thus, the RERO and the suggested addition should be able to protect Simon and Speck against permanent and transient faults and make the DFA attacks presented in [37–39] more difficult; however, we do not claim that it will be able to detect all types of DFA 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); is diverse from an architecture based on robust codes which can detect (with probability) any error. These two solutions have two different goals, the first gives reliability and the second gives hardware security (against DFA).

Finally, It is noted that according to [41], the linear compressor can make the code not robust anymore. Furthermore, this compressor is not required at all since cubic function can be designed for any vector length. In the context of its hardware overhead, there are high rate robust codes [42] that have lower hardware complexity [43].

## 4.4 Comparison with Previous Work

There has not been any prior work done on error detection methods for these ciphers to the best of our knowledge. In [44], the authors present fault diagnosis of Pomaranch cipher. They have used bit-interleaved scheme for error detection. We compare the overheads of Pomaranch with the proposed scheme. The area and throughput overheads for Pomaranch are 21% and 12%, respectively. The proposed schemes have area and throughput overheads of 30% and 10% for Simon and 11% and 6% for Speck, respectively. Since the architecture of Pomaranch and presented fault detection scheme is a lot different than the proposed method, the differences in the overheads are reasonably justified. The proposed fault detection methods can be applied to Pomaranch and other ciphers as well to obtain approximately closer overheads.

# Chapter 5

# Conclusion

Our research group has extensive experience in cryptographic engineering and fault diagnosis as well as editing IEEE Transactions journals [45–91]. This thesis proposes reliable and efficient error detection architectures for the block ciphers Simon and Speck. The proposed schemes are optimized for low-area and low-power applications since Simon and Speck are among lightweight block ciphers. We propose diagnosis approaches for inner sub-blocks of these ciphers and present an approach for alleviating the throughput overheads. The simulation results show that the proposed error detection schemes can detect close to 100% of the injected faults. We have also implemented our proposed architectures on two Xilinx FPGA families, i.e., Zynq-7000 and Virtex-7 families. The implementation results show that the power, area, and delay overheads incurred by the proposed architectures are acceptable. Therefore, the proposed architectures for Simon and Speck block ciphers can be reliably and efficiently used and further tailored by customizing the architectures based upon the requirements in terms of reliability, security, and overhead tolerance.

## 5.1 Future Works

The proposed work provides fault detection in the inner sub-blocks of the cipher and can protect against suprious faults. The main objective of the proposed work is towards achieving a low-area and low-power consuming architecture along with 100% fault detection.

The fault detection scheme used in this proposal was RERO. However, previous works have been performed on other lightweight block ciphers like CLEFIA and PRESENT using different fault detection schemes. These methods can also be used to test SIMON and SPECK and the performance metrics can be obsereved.

An important parameter while implementing a design is the type of platform used which determines the hardware fabric that will be utilized for implementation. The proposed work was implemented on Xilinx Zynq and Virtex FPGA families. However, the proposed design can be implemented on other FPGA devices and the results can be benchmarked. Similarly, compared to an FPGA, implementations on ASIC platforms might be more suitable in terms of area and power optimizations for some applications. Therefore, ASIC implementation can be the next part of the proposed work.

The proposed work has been proved to be resilient against natural faults. It has been also shown that a number of fault attacks can be thwarted based on the error models used in this thesis. This architecture can be further modified to provide better protection against such attacks. Thus, in the future, a complete solution against all types of attacks can be devised to devise an immune block cipher.

# References


[1] A. Moradi, A. Poschmann, S. Ling, C. Paar, and H. Wang, "Pushing the limits: A very compact and a threshold implementation of AES," *Proc. Advances in Cryptology*, pp. 69 – 88, 2011.

[2] C. D. CanniÃšre, O. Dunkelman, and M. Knezevic, "KATAN and KTANTAN - a family of small and efficient hardware-oriented block ciphers," *Proc. Cryptographic Hardware and Embedded Systems*, no. 272 - 288, 2009.

[3] K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, and T. Shirai, "Piccolo: An ultra-lightweight blockcipher," *Proc. Cryptographic Hardware and Embedded Systems*, pp. 342 – 357, 2011.

[4] F. X. Standaert, G. Piret, N. Gershenfeld, and J. J. Quisquater, "SEA: A scalable encryption algorithm for small embedded applications," *Proc. Smart Card Research and Advanced Applications*, pp. 222 – 236, 2006.

[5] J. Guo, T. Peyrin, A. Poschmann, and M. J. B. Robshaw, "The LED block cipher," *Proc. Cryptographic Hardware and Embedded Systems*, pp. 326 – 341, 2011.

[6] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoel, "PRESENT: An ultra-lightweight block-cipher," *Proc. Cryptographic Hardware and Embedded Systems*, pp. 450 – 466, 2007.

[7] Beaulieu, Ray, Shors, Douglas, Smith, Jason, Treatman-Clark, Stefan, Weeks, Bryan, Wingers, and Louis, "The SIMON and SPECK families of block ciphers," *Proceedings of the 52nd Annual Design Automation Conference*, p. 175, 2015.

[8] Beaulieu, Ray, Shors, Douglas, Smith, Jason, Treatman-Clark, Stefan, Weeks, Bryan, Wingers, and Louis, "SIMON and SPECK: Block ciphers for the internet of things," *NIST Lightweight Cryptography Workshop*, 2015.

[9] A. Biryukov, A. Roy, and V. Velichkov, "Differential analysis of block ciphers SIMON and SPECK," *Proc. Fast Software Encryption*, pp. 546 – 570, 2014.

[10] S. Sun, L. Hu, P. Wang, K. Qiao, X. Ma, and L. Song, "Automatic security evaluation and (related-key) differential characteristic search: Application to SIMON, PRESENT, LBLOCK, DES(L) and other bit-oriented block ciphers," *Proc. Advances in Cryptology*, pp. 158–178, 2014.

[11] H. Tupsamudre, S. Bisht, and D. Mukhopadhyay, "Differential fault analysis on the families of SIMON and SPECK ciphers," *Proc. Fault Diagnosis and Tolerance in Cryptography*, pp. 40–48, 2014.

[12] 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, June 2006.

[13] G. D. Natale, M. Doulcier, M. L. Flottes, and B. Rouzeyre, "A reliable architecture for parallel implementations of the Advanced Encryption Standard," *J. Electronic Testing: Theory and Applications*, vol. 25, no. 4, pp. 269 – 278, Aug. 2009.

[14] 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.

[15] M. Mozaffari Kermani and R. Azarderakhsh, "Efficient fault diagnosis schemes for reliable lightweight cryptographic ISO/IEC standard CLEFIA benchmarked on ASIC and FPGA," *IEEE Trans. Ind. Electron.*, vol. 60, no. 12, pp. 5925–5932, Dec. 2013.

[16] 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, 2008.

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

[18] M. Yasin, B. Mazumdar, S. S. Ali, and O. Sinanoglu, "Security analysis of logic encryption against the most effective side-channel attack: DPA," *Proc. DFTS*, pp. 97–102, 2015.

[19] D. Karaklajic, J.-M. Schmidt, and I. Verbauwhede, "Hardware designer's guide to fault attacks," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 12, pp. 2295 – 2306, 2013.

[20] R. Karri, G. Kuznetsov, and M. Goessel, "Parity-based concurrent error detection of substitution-permutation network block ciphers," *Proc. Cryptographic Hardware and Embedded Systems*, pp. 113–124, 2003.

[21] M. Karpovsky, K. J. Kulikowski, and A. Taubin, "Robust protection against fault-injection attacks on smart cards implementing the Advanced Encryption Standard," *Proc. Dependable Systems and Networks.*, pp. 93–101, 2004.

[22] K. Wu and R. Karri, "Algorithm-level recomputing with shifted operands-A register transfer level concurrent error detection technique," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 3, pp. 413–422, 2006.

[23] J. H. Patel and L. Y. Fung, "Concurrent error detection in ALUs by recomputing with shifted operands," *IEEE Trans. Comput.*, vol. C-31, no. 7, pp. 589–595, 1982.

[24] J. Li and E. E. Swartzlander, "Concurrent error detection in ALUs by recomputing with rotated operands," *Proc. Defect and Fault Tolerance in VLSI Systems*, pp. 109–116, 1992.

[25] X. Guo and R. Karri, "Recomputing with permuted operands: A concurrent error detection approach," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 10, pp. 1595–1608, 2013.

[26] R. Karri, K. Wu, P. Mishra, and Y. Kim, "Concurrent error detection schemes of fault based side-channel cryptanalysis of symmetric block ciphers," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, no. 12, pp. 1509 – 1517, 2002.

[27] A. Satoh, T. Sugawara, N. Homma, and T. Aoki, "High-performance concurrent error detection scheme for AES hardware," *Proc. CHES.*, pp. 110–112, 2008.

[28] J. Rajendran, H. Borad, S. Mantravadi, and R. Karri, "SLICED: Slide based concurrent error detection technique for symmetric block cipher," *Proc. HOST*, pp. 70–75, 2010.

[29] M. Karpovsky, K. Kulikowski, and A. Taubin, *Differential Fault Analysis Attack Resistant Architectures for the Advanced Encryption Standard*. Springer US, 2004.

[30] D. Vasudevan, P. Lala, and J. Parkerson, "Self-checking carry-select adder design based on two-rail encoding," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, no. 12, pp. 2696 – 2705, Dec. 2007.

[31] M. Nicolaidis, "Carry checking/parity prediction adders and ALUs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 1, pp. 121 – 128, 2003.

[32] A. Tenca and M. Ercegovac, "A variable long-precision arithmetic unit design for recon-

figurable coprocessor architectures," *Proc. IEEE Symp. FPGAs for Custom Computing Machines*, pp. 216 – 225, April 1998.

[33] G. Xiaofei and R. Karri, "Recomputing with permuted operands: A concurrent error detection approach," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 32, no. 10, pp. 1595 – 1608, Sep. 2013.

[34] S. Durand and C. Piguet, "FPGA with selfrepair capabilities," *Int. ACM/SIGDA Workshop on Field Programmable Gate Arrays*, pp. 1 – 10, Feb. 1994.

[35] J. Emmert, C. Stroud, and M. Abramovici, "Online fault tolerance for FPGA logic blocks," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 2, pp. 216 – 226, Feb. 2007.

[36] M. A. Akbar and J. A. Lee, "Comments on self-checking carry-select adder design based on two-rail encoding," *IEEE Trans. Circuits Syst. I*, vol. 61, no. 7, pp. 2212–2214, 2014.

[37] H. Tupsamudre, S. Bisht, and D. Mukhopadhyay, "Differential fault analysis on the families of SIMON and SPECK ciphers," *Proc. Fault Diagnosis and Tolerance in Cryptography*, pp. 40–48, 2014.

[38] J. Takahashi and T. Fukunaga, "Fault analysis on SIMON family of lightweight block ciphers," *Proc. Information Security and Cryptology*, pp. 175–189, 2015.

[39] J. C. G. VÂŽasquez, F. Borges, R. Portugal, and P. Lara, "An efficient one-bit model for differential fault analysis on SIMON family," *Proc. Fault Diagnosis and Tolerance in Cryptography*, pp. 61–70, 2015.

[40] A. Barenghi, L. Breveglieri, I. Koren, and D. Naccache, "Fault injection attacks on cryptographic devices: Theory, practice and countermeasures," *Proc. the IEEE*, vol. 100, no. 11, pp. 3056–3076, 2012.

[41] M. Karpovsky and Taubin, "New class of nonlinear systematic error detecting codes," *IEEE Trans. Information Theory*, vol. 50, no. 8, pp. 1818–1819.

[42] V. Tomashevich, Y. Neumeier, R. Kumar, O. Keren, and I. Polian, "Protecting cryptographic hardware against malicious attacks by nonlinear robust codes," *Proc. Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, pp. 40–45, 2014.

[43] Y. Neumeier, Y. Pesso, and O. Keren, "Efficient implementation of punctured parallel finite field multipliers," *IEEE Trans. Circuits and Systems I: Regular Papers*, vol. 62, no. 9, pp. 2260–2267, 2015.

[44] M. Mozaffari Kermani, R. Azarderakhsh, and A. Aghaie, "Reliable and error detection architectures of Pomaranch for false-alarm-sensitive cryptographic applications," *IEEE Trans. VLSI Systems*, vol. 23, no. 12, pp. 2804–2812, 2015.

[45] M. Mozaffari-Kermani, E. Savas, and S. Upadhyaya, "Guest editorial: Introduction to the special issue on emerging security trends for deeply-embedded computing systems," *IEEE Transactions on Emerging Topics in Computing*, 2016.

[46] R. Azarderakhsh and M. Mozaffari-Kermani, "High-performance two-dimensional finite field multiplication and exponentiation for cryptographic applications," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 34, no. 10, pp. 1569 – 1576, Oct. 2015.

[47] M. Mozaffari-Kermani, R. Azarderakhsh, S. Bayat-Sarmadi, and C. Lee, "Systolic gaussian normal basis multiplier architectures suitable for high-performance applications," vol. 23, no. 9, pp. 1969–1972, Sept. 2015.

[48] A. Mohsen-nia, M. Mozaffari-Kermani, S. Sur-kolay, A. Raghunathan, and N. K. Jha, "Energy-efficient long-term continuous personal health monitoring," *IEEE Trans. Multi-Scale Computing Systems,*, vol. 1, no. 2, April 2015.

[49] M. Mozaffari-Kermani, N. Manoharan, and R. Azarderakhsh, "Reliable Radix-4 complex division for fault-sensitive applications," *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.*, vol. 34, no. 4, pp. 656 – 667, April 2015.

[50] R. Azarderakhsh, M. Mozaffari-Kermani, and K. U. Jarvinen, "Secure and efficient architectures for single exponentiation in finite field suitable for high-performance cryptographic applications," *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.*, vol. 34, no. 3, pp. 332 – 340, Mar. 2015.

[51] M. Mozaffari-Kermani, S. Bayat-Sarmadi, and R. Azarderakhsh, "Fault-resilient lightweight cryptographic block ciphers for secure embedded systems," *IEEE Embedded Sys.*, vol. 6, no. 4, pp. 89 – 92, Dec. 2014.

[52] S. Bayat-Sarmadi, M. Mozaffari-Kermani, and A. Reyhani-Masoleh, "Efficient and concurrent reliable realization of the secure cryptographic SHA-3 algorithm," *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.*, vol. 33, no. 7, pp. 1105 – 1109, Jul. 2014.

[53] J. Pan, R. Azarderakhsh, M. Mozaffari-Kermani, C. Lee, C. Chiou, and J. Lin, "Low latency digit-serial systolic double basis multiplier over $GF(2^m)$ using subquadratic toeplitz matrix-vector product approach," *IEEE Trans. Comput.*, vol. 63, no. 5, pp. 1169 – 1181, May 2014.

[54] K. U. Jarvinen, R. Azarderakhsh, and M. Mozaffari-Kermani, "Efficient algorithm and architecture for elliptic curve cryptography for extremely constrained secure applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 4, pp. 1144 – 1155, Apr. 2014.

[55] S. Bayat-Sarmadi, M. Mozaffari-Kermani, R. Azarderakhsh, and C. Lee, "Dual basis super-serial multipliers for secure applications and lightweight cryptographic architec-

tures," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 61, no. 2, pp. 125 – 129, Feb. 2014.

[56] M. Mozaffari-Kermani, R. Azarderakhsh, K. Ren, and J. Beuchat, "Guest editorial: Introduction to the special issue on emerging security trends for biomedical computations, devices, and infrastructures," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2016.

[57] M. Mozaffari-Kermani and R. Azarderakhsh, "Efficient fault diagnosis schemes for reliable lightweight cryptographic ISO/IEC standard CLEFIA benchmarked on ASIC and FPGA," *IEEE Trans. Ind. Electron.*, vol. 60, no. 12, pp. 5925 – 5932, Dec. 2013.

[58] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "Efficient and high-performance parallel hardware architectures for the AES-GCM," *IEEE Trans. Comput.*, vol. 61, no. 8, pp. 1165 – 1178, Aug. 2013.

[59] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "A low-power high-performance concurrent fault detection approach for the composite field S-box and inverse S-box," *IEEE Trans. Comput.*, vol. 60, no. 9, pp. 1327 – 1340, Sept. 2011.

[60] 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.

[61] 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.

[62] M. Mozaffari Kermani and A. Reyhani Masoleh, "Fault detection structures of the S-boxes and the inverse S-boxes for the Advanced Encryption Standard," *J. Electronic Testing: Theory and Applications (JETTA)*, vol. 25, no. 4, pp. 225 – 245, Aug. 2009.

[63] M. Mozaffari-Kermani, R. Azarderakhsh, and J. Xie, "Error detection reliable architectures of Camellia block cipher applicable to different variants of its substitution boxes," *Proc. IEEE Asian Hardware Oriented Security and Trust Symposium (HOST)*, to appear in 2016.

[64] B. Koziel, R. Azarderakhsh, A. Jalali, D. Jao, and M. Mozaffari-Kermani, "NEON-SIDH: Efficient implementation of supersingular isogeny diffie-hellman key exchange protocol on ARM," *Proc. Conf. Cryptology and Network Security (CANS)*, to appear in 2016.

[65] B. Koziel, R. Azarderakhsh, and M. Mozaffari-Kermani, "Fast hardware architectures for supersingular isogeny diffie-hellman key exchange on FPGA," *Proc. Int. Conf. (INDOCRYPT)*, to appear in 2016.

[66] M. Mozaffari-Kermani and R. Azarderakhsh, "Lightweight hardware architectures for fault diagnosis schemes of efficiently-maskable cryptographic substitution boxes," *Proc. IEEE Int. Conf. ICECS*, to appear in 2016.

[67] 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 Computing Syst. (special issue on Embedded Device Forensics and Security: State of the Art Advances)*, to appear in 2016.

[68] A. Aghaie, M. Mozaffari-Kermani, and R. Azarderakhsh, "Fault diagnosis schemes for secure lightweight cryptographic block cipher RECTANGLE benchmarked on FPGA," *Proc. IEEE Int. Conf. ICECS*, to appear in 2016.

[69] M. Mozaffari-Kermani, R. Azarderakhsh, and M. Mirakhorli, "Multidisciplinary approaches and challenges in integrating emerging medical devices security research and education," *Proc. Conf. American Society for Engineering Education*, to appear in 2016.

[70] R. Ramadoss, M. Mozaffari-Kermani, and R. Azarderakhsh, "Efficient error detection architectures for CORDIC through recomputing with encoded operands," *Proc. IEEE Int. Symp. Circuits and Systems (ISCAS)*, pp. 2154 – 2157, May 2016.

[71] B. Koziel, M. Mozaffari-Kermani, R. Azarderakhsh, and D. Jao, "Post-quantum cryptography on FPGA based on isogenies on elliptic curves," *Proc. eprint 2016/672*, pp. 1 – 18, 2016.

[72] B. Koziel, A. Jalali, M. Mozaffari-Kermani, R. Azarderakhsh, and D. Jao, "NEON-SIDH: Efficient implementation of supersingular isogeny diffie-hellman key-exchange protocol on ARM," *Proc. eprint 2016/669*, pp. 1 – 16, 2016.

[73] B. Koziel, M. Mozaffari-Kermani, and R. Azarderakhsh, "Low-resource and fast binary edwards curves cryptography using gaussian normal basis," *Proc. Int. Conf. (INDOCRYPT)*, pp. 347 – 369, 2015.

[74] M. Mozaffari-Kermani and R. Azarderakhsh, "Reliable hash trees for post-quantum stateless cryptographic hash-based signatures," *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems (DFT)*, pp. 103 – 108, Oct. 2015.

[75] M. Mozaffari-Kermani and R. Azarderakhsh, "Integrating emerging cryptographic engineering research and security education," *Proc. Conf. American Society for Engineering Education*, pp. 1 – 13, Jun. 2015.

[76] C. E. Kennedy and M. Mozaffari-Kermani, "Generalized parallel crc computation on FPGA," *Proc. IEEE Conf. Elec. Comput. Eng.*, pp. 107 – 113, May. 2015.

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

[78] B. Koziel, R. Azarderakhsh, and M. Mozaffari-Kermani, "Post-quantum cryptography

on FPGA based on isogenies on elliptic curves," *IEEE Trans. Circuits Syst. I, Reg. Papers*, to appear in 2016.

[79] M. Zhang, M. Mozaffari-Kermani, A. Raghunathan, and N. K. Jha, "Energy-efficient and secure sensor data transmission using encompression," *Proc. IEEE Int. Conf. VLSI Design*, pp. 31 – 36, Jan. 2013.

[80] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "Reliable hardware architectures for the third-round SHA-3 finalist grostl benchmarked on FPGA platform," *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems (DFT)*, pp. 325 – 331, Oct. 2011.

[81] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "A high-performance fault diagnosis approach for the AES SubBytes utilizing mixed bases," *Proc. IEEE Workshop Fault Diagnosis and Tolerance in Cryptography (FDTC)*, pp. 80 – 87, Sep. 2011.

[82] M. Mozaffari Kermani and A. Reyhani Masoleh, "A low-cost S-box for the Advanced Encryption Standard using normal basis," *Proc. IEEE Int. Conf. Electro/Information Technology (EIT)*, pp. 52 – 55, Jun. 2009.

[83] M. Mozaffari-Kermani and A. Reyhani-Masoleh , "A lightweight concurrent fault detection scheme for the AES S-Boxes using normal basis," *Proc. LNCS Cryptographic Hardware and Embedded Systems (CHES)*, pp. 113 – 129, Aug. 2008.

[84] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "A structure-independent approach for fault detection hardware implementations of the Advanced Encryption Standard," *Proc. IEEE Workshop Fault Diagnosis and Tolerance in Cryptography (FDTC)*, pp. 47 – 53, Sep. 2007.

[85] M. Mozaffari Kermani and A. Reyhani Masoleh, "Parity-based fault detection architecture of S-box for Advanced Encryption Standard," *Proc. IEEE Int. Symp. Defect and Fault Tolerance in VLSI Systems (DFT)*, pp. 572 – 580, Oct. 2006.

[86] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "Parity prediction of S-box for AES," *Proc. IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 2357 – 2360, May 2006.

[87] M. Mozaffari-Kermani and R. Azarderakhsh and V. Singh, "Reliable low-latency Viterbi algorithm architectures benchmarked on ASIC and FPGA," *IEEE Trans. Circuits Syst. I, Reg. Papers*, to appear in 2016.

[88] M. Mozaffari-Kermani, R. Azarderakhsh, and V. Singh, "Reliable low-latency Viterbi algorithm architectures benchmarked on ASIC and FPGA," *IEEE Trans. Circuits Syst. I, Reg. Papers*, to appear in 2016.

[89] P. Chen, S. N. Basha, M. Mozaffari-Kermani, R. Azarderakhsh, and J. Xie, "FPGA realization of low register systolic all-one-polynomial multipliers over $GF(2^m)$ and their applications in trinomial multipliers," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, to appear in 2016.

[90] A. Aghaie, M. Mozaffari-Kermani, and R. Azarderakhsh, "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.

[91] M. Mozaffari-Kermani, S. Sur-kolay, A. Raghunathan, and N. K. Jha, "Systematic poisoning attacks on and defenses for machine learning in healthcare," *IEEE J. Biomedical and Health Informatics*, vol. 19, no. 6, pp. 1893 – 1905, Nov. 2015.