Innovative Fault Detection for AES in Embedded Systems: Advancing Resilient and Sustainable Digital Security

Hassen Mestiri
Department of Computer Engineering, College of Computer Engineering and Sciences, Prince Sattam bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia
h.mestiri@psau.edu.sa (corresponding author)

Imen Barraj
Department of Computer Engineering, College of Computer Engineering and Sciences, Prince Sattam bin Abdulaziz University, Al-Kharj 11942, Saudi Arabia
i.barraj@psau.edu.sa

Mohsen Machhout
Electronics and Micro-Electronics Laboratory, Faculty of Sciences of Monastir, University of Monastir, Tunisia
mohsen.machhout@fsm.rnu.tn

Received: 6 December 2024 | Revised: 27 December 2024 | Accepted: 12 January 2025

 DOI: https://doi.org/10.48084/etasr.9852

ABSTRACT

The AES algorithm is commonly used in embedded systems for security purposes, but its robustness can be compromised by natural and malicious faults, leading to potential information leakage. Various fault detection schemes have been proposed to protect it against differential fault analysis attacks. These schemes aim to detect and mitigate any potential vulnerabilities in the AES algorithm, ensuring system security. The implementation of fault detection schemes aligns with Sustainable Development Goal (SDG) 9, which focuses on building resilient infrastructure and promoting inclusive and sustainable industrialization. Enhancing the security of embedded systems through these measures contributes to creating a more secure and sustainable digital environment for all. This study introduces a new fault-parity detection scheme that involves comparing the correct parity of the rounded output with the predicted parity based on AES processing steps. The strengths and weaknesses of this scheme in defending against fault attacks are also discussed. The experimental results demonstrate that the proposed fault detection scheme achieves an impressive fault coverage of 99.999%. Implemented on the Xilinx Virtex-5 FPGA, the scheme was compared to existing methods in terms of fault coverage, area overhead, frequency degradation, and throughput. These results highlight the ability of the proposed scheme to strike a balance between implementation cost and AES security.

Keywords-security; cryptography; fault attacks; encryption algorithm; secure communication

I. INTRODUCTION

The AES algorithm, standardized by NIST in 2001, has sparked interest in developing efficient hardware implementations for diverse applications [1]. Evaluations of these implementations using ASIC and FPGA libraries have shown promising results [2-5]. Today, AES is widely utilized in telecommunications and financial transactions due to its robust security features. Furthermore, it is crucial to protect the AES algorithm from potential threats such as power analysis attacks that exploit power consumption patterns and fault injection attacks that aim to disrupt the algorithm's operation.

Fault injection attacks are a potent method for compromising cryptographic algorithms that lack adequate protection measures. By introducing faults into the execution process of a cryptographic implementation, attackers can exploit the resulting errors to gain insight into the encryption key being used. These attacks can be particularly devastating as they allow malicious actors to bypass security measures and access sensitive information [3-8]. Several fault detection schemes, such as redundancy-based schemes or error detection codes, have been proposed to provide reliable implementations against fault attacks [9-12].

In [9], an efficient fault detection method was introduced for the AES algorithm, dividing the round architecture into three parts and strategically inserting two pipeline registers. Simulations showed improved fault detection capabilities, indicating the potential for improving security measures in cryptographic systems. In [10], an AES encryption/decryption system was developed for IoT infrastructure and resource-constrained applications, using a 32-bit architecture. LC-FRAES was proposed, a fault-resilient architecture that optimizes resource sharing between encryption and decryption processes, enhancing data-path efficiency and on-the-fly key expansion unit performance. In [1], a new fault detection scheme was proposed to improve security by dividing the AES 32-bit round into two half rounds and implementing input and pipeline registers between them. This adaptability makes the scheme suitable for securing both pipeline and iterative architectures. In [11], fault-tolerant designs were analyzed for cryptographic applications, demonstrating the effectiveness of R-CFTA+ and HT-CFTA+ in high-security scenarios and comparing their performance with related works, using system efficiency as a key design metric.

This study introduces a fault detection scheme to enhance the security of the AES algorithm against fault injection attacks. The vulnerabilities and advantages of this scheme are thoroughly analyzed, and insights into its FPGA implementation results are provided. The proposed fault detection scheme aims to contribute to the development of secure and reliable cryptographic systems.

II. BACKGROUNDS

The AES encryption algorithm operates in rounds, with their number varying based on the key length: 10 rounds for a 128-bit key, 12 rounds for a 192-bit key, and 14 rounds for a 256-bit key. During each round, except for the final one, four transformations are applied to the data block: SubBytes, ShiftRows, MixColumns, and AddRoundKey. The round key used in each round is derived from the initial key through a dedicated key scheduling process.

- SubBytes involves a non-linear substitution applied to each byte based on a lookup table.
- ShiftRows involves shifting each row of the state by a certain number of positions.
- MixColumns combines the bytes in each column using a linear transformation.
- AddRoundKey XORs each byte with a round key derived from the cipher key through a key schedule.

III. FAULT DETECTION SCHEME DESIGN

Until now, error detection systems presented in the literature have allowed the hardware implementation of AES. Although these systems achieve a very high detection rate against fault injection attacks, they only serve to protect the four AES transformations: SubBytes, ShiftRows, MixColumns, and AddRoundKey. This study proposes a method to protect the entire hardware implementation of AES: the initial round, the four AES transformations, the key scheduler, the controller, and the various multiplexers and state registers. This method is based on the principle of hybrid redundancy. Figure 1 shows the architecture of the proposed system.

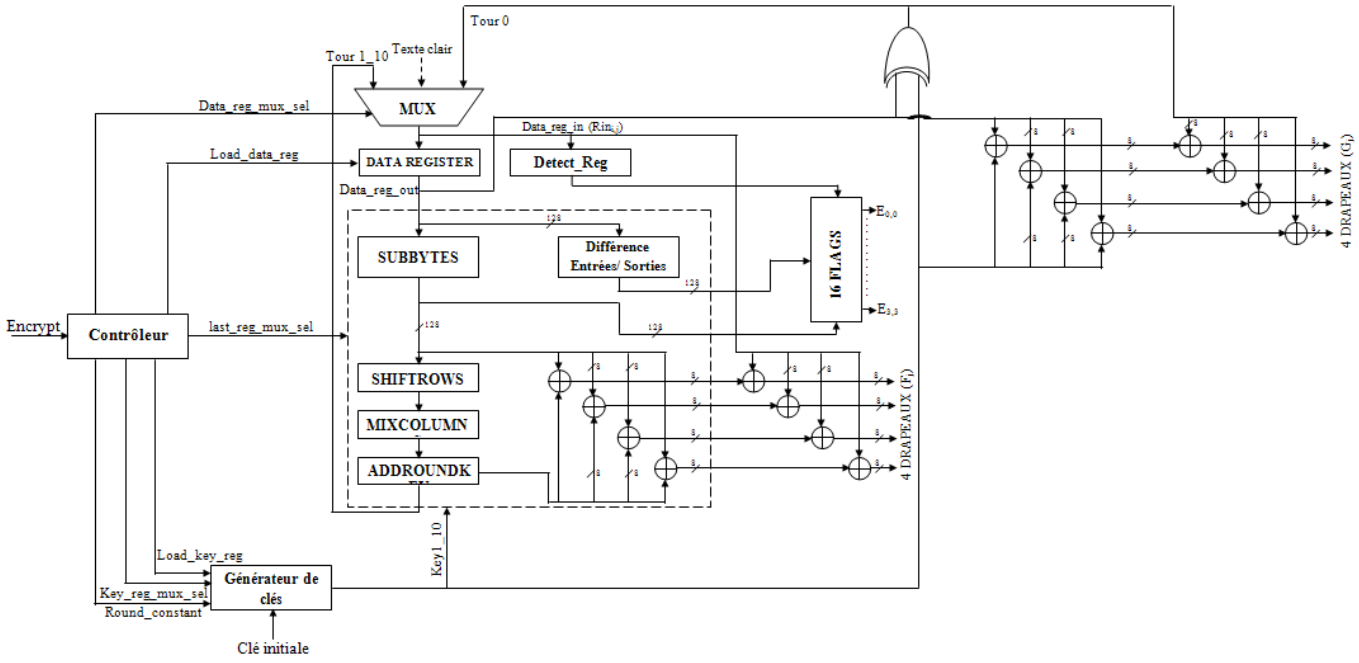

Fig. 1. Fault detection scheme for SubBytes and state register.

The proposed system utilizes the parity technique to add redundancy to the key scheduler, controller, and state registers. The parity technique involves adding an extra bit to each byte of data to ensure that the number of ones in the byte is always even. This helps detect any single-bit errors that may occur during data transmission or processing. Overall, the hybrid redundancy approach enhances the security and reliability of the AES hardware implementation by introducing error detection capabilities in critical components, thus mitigating the risk of successful attacks.

A. SubByte Transformation and State Register Protection

The SubBytes transformation in hardware involves 16 S-Boxes, each corresponding to a byte in a 128-bit message. Figure 2 shows the fault detection scheme for SubBytes and the state register.

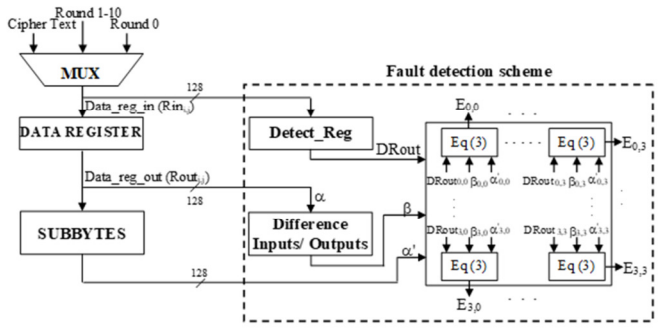

Fig. 2. Fault detection scheme for SubBytes and state register.

As seen in Figure 2, the difference β_i is determined by subtracting the input α_i from the output α_i' of each S-Box block. To streamline this process, a table containing the pre-calculated 256 possible values of β is utilized for efficient computation. The difference β is obtained through addition modulo 2 between the input α_i and the output α_i'.

$$\beta_{i,j} = \alpha_{i,j} \oplus \alpha'_{i,j} \quad (1)$$

with $0 \leq i, j \leq 3$, $\beta_{i,j}$, $\alpha_{i,j}$ and $\alpha'_{i,j}, \in GF(2^8)$,

To enhance the security of the data register against fault attacks, the Detect_Reg status register is introduced, with a size of 128 bits. As the SubBytes transformation takes on the output of the data register as input, any discrepancies can be easily identified and addressed by comparing the values in both registers.

$$\beta_{i,j} = Rout_{i,j} \oplus \alpha'_{i,j} \quad (2)$$

with $0 \leq i, j \leq 3$ and $Rout_{i,j} \in GF(2^8)$,

The 16 error detection flags ($E_{1,1}$ to $E_{3,3}$ are crucial in identifying errors within the data register and SubBytes function. By comparing the difference β with the modulo 2 addition between the output of the Detect_Reg status register and the SubBytes transformation, these flags provide a reliable method for detecting errors in the system. This process helps ensure data integrity and security in cryptographic systems.

$$E_{i,j} = \beta_{i,j} \oplus DRout_{i,j} \oplus \alpha'_{i,j} \quad (3)$$

with $0 \leq i, j \leq 3$, $E_{i,j}$ and $DRout_{i,j} \in GF(2^8)$.

The proposed detection system is independent of how the SubBytes and Inv_SubBytes transformations are implemented (LUT or combinational logic). This allows for flexibility in the design and implementation of these transformations, making the system adaptable to different architectures and requirements.

B. Protection of the Multiplexer and Linear Transformations

The ShiftRows transformation takes as input the output of the SubBytes transformation. As the ShiftRows transformation shifts the bytes without altering their values, the parities of the input and output remain unchanged. The hardware implementation of the MixColumns transformation is more intricate than that of the ShiftRows function because each byte of the input message affects four bytes of the output. Given that each column of the state matrix is multiplied by a vector $\{03, 02, 01, 01\}$ whose modulo 2 addition between its elements equals 1, the output signatures of the ShiftRows and MixColumns transformations are equivalent, as demonstrated by the following system:

$$\begin{aligned} P_{SRout_j} &= s_{0,j} \oplus s_{1,j} \oplus s_{2,j} \oplus s_{3,j} \\ P_{MC_j} &= s'_{0,j} \oplus s'_{1,j} \oplus s'_{2,j} \oplus s'_{3,j} \\ &= (02 \oplus 03 \oplus 01 \oplus 01)(s_{0,j} \oplus s_{1,j} \oplus s_{2,j} \oplus s_{3,j}) \quad (4) \\ &= s_{0,j} \oplus s_{1,j} \oplus s_{2,j} \oplus s_{3,j} \\ \Rightarrow P_{MC_j} &= P_{SRout_j} \end{aligned}$$

with P_{SRout_j} and P_{MC_j} being the parities of the output of the j^{th} column of ShiftRows and MixColumns respectively.

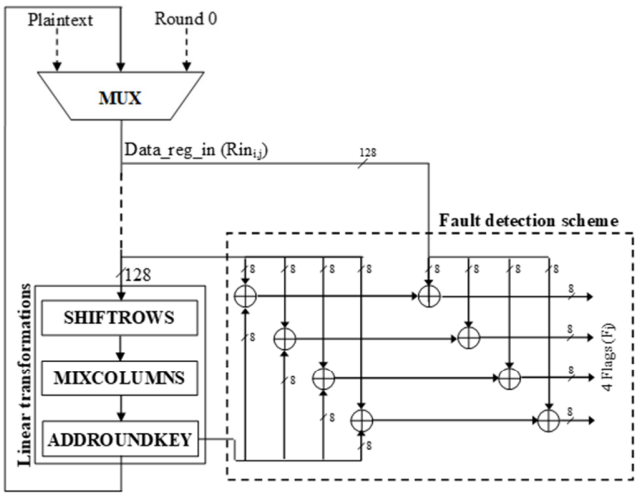

Fig. 3. Fault detection scheme for linear transformations and multiplexer.

The AddRoundKey transformation computes the modulo 2 addition between the output of the MixColumns transformation and the round key to obtain the AES round output. Consequently, the parity of the output equals the modulo 2 addition between the parity of the round key and that of the

MixColumns output. Since AddRoundKey is the only linear transformation that affects parity, according to (5), the signature of the three linear transformations can be calculated as:

$$\begin{aligned} P_{out_j} &= k_{0,j} \oplus k_{1,j} \oplus k_{2,j} \oplus k_{3,j} \oplus s_{0,j} \oplus s_{1,(j+1)\bmod 4} \\ &\quad \oplus s_{2,(j+2)\bmod 4} \oplus s_{3,(j+3)\bmod 4} \\ &= P_{K_j} \oplus P_{SRin_j} \end{aligned} \quad (5)$$

with P_{out_j}, P_{K_j}, and P_{SRin_j} being the parity of the output of the round, the parity of the key schedule, and the parity of the input of ShiftRows, respectively.

The multiplexer acts as a key component in the encryption process by efficiently managing the data flow between each round of AES. This streamlined transfer process helps maintain the integrity and coherence of the encrypted output, ensuring a secure and reliable encryption algorithm.

$$P_{out_j} = P_{Rin_j} = \sum_{i=0}^{3} Rin_{i,j} \quad (6)$$

The error detection flags F_j are generated through a comparison of the multiplexer output with the modulo 2 addition of the ShiftRows function input and the key round. This process is executed for each 32-bit column utilizing 256 2-input XOR gates. The four error detection flags F_j serve to identify errors that may arise or be introduced into the multiplexer and the three linear functions of AES. The F_j flags are calculated as follows:

$$F_j = P_{Rin_j} \oplus P_{K_j} \oplus P_{SRin_j} \quad (7)$$

C. Initial Round Protection

During the initial encryption round, the plaintext undergoes a modulo 2 addition with the initial key. The multiplexer then receives the output of this initial round as input. To protect against natural and malicious fault attacks during the execution of the initial round, a signature is calculated using the outputs of the data register, the key generator, and the initial round (see Figure 1). The detection procedure is applied to every four bytes of the AES data message, resulting in the generation of four error detection flags. This necessitates an increase of 256 2-input XOR gates in the hardware implementation. The flags are calculated as follows:

$$G_j = \sum_{i=0}^{3} DRout_{i,j} \oplus \sum_{i=0}^{3} Key_{i,j} \oplus \sum_{i=0}^{3} Rin_{i,j} \quad (8)$$

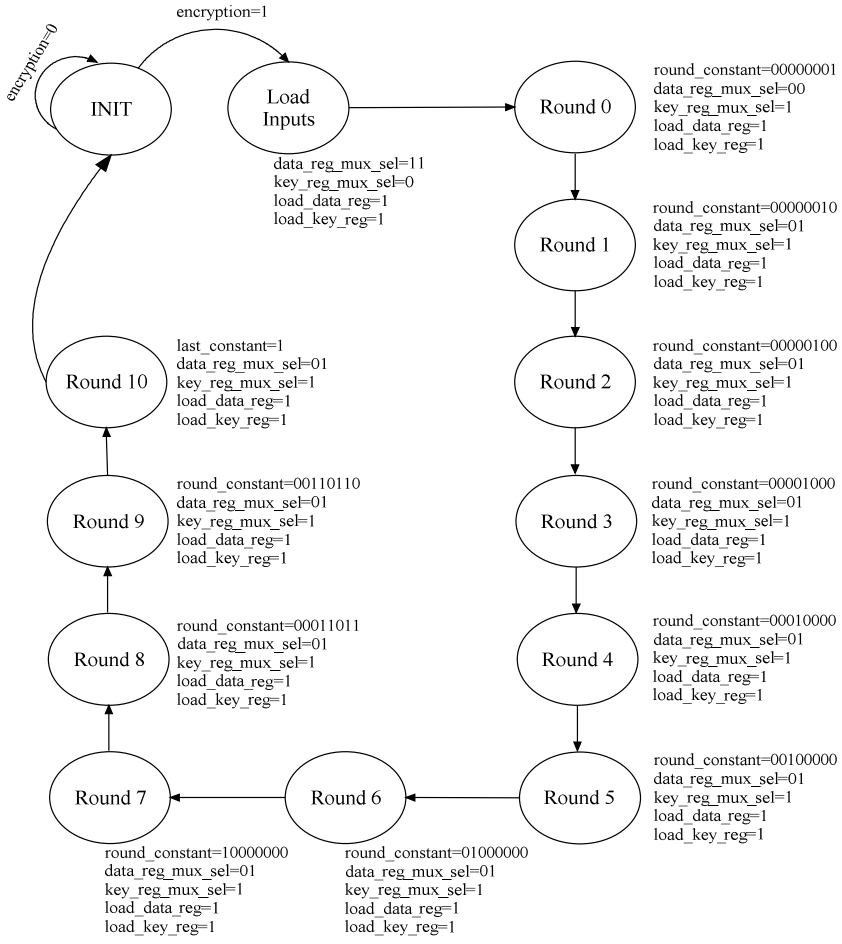

Fig. 4. Controller's organizational chart

D. Controller Protection

The AES controller is a state controller that generates control signals for the various hardware implementation modules. This controller includes a Finite State Machine (FSM) that manages a set of states and synchronization signals. The FSM is based on the principle of one-hot encoding. Figure 4 shows the controller's flowchart. First, we began by identifying the various vulnerable elements of the controller to fault injection attacks. Transient faults (single and multiple) that last one or more clock cycles were considered.

The perturbation of a single flip-flop's behavior was examined. This was achieved by generating a transient pulse at the input of the exclusive OR logic gate, as shown in Figure 5. The transient pulse reaches the flip-flop input based on the length of the corresponding propagation path. If the pulse arrives at the input during the active edge of the system clock, it will be recorded as an erroneous value. If this pulse results from the transition of the flip-flop output from state 1 to state 0 for a duration shorter than one clock cycle, the finite state machine becomes blocked, leading the AES to run in an infinite loop (see Figure 6). On the contrary, when the flip-flop output is set to 1 for a duration of one clock cycle, two states are triggered concurrently. This disruption in the controller's behavior causes a variation in the number of execution cycles. For instance, if faults affect the hardware implementation of the ninth state flip-flop during the second round of AES, the algorithm will perform the tenth round and produce an erroneous encrypted text. Regardless of the duration of the faults injected into the state flip-flops, the AES algorithm generates erroneous and exploitable ciphertext, rendering it vulnerable to fault attacks.

To protect the controller, a second FSM was implemented based on the principle of one-hot encoding. This one consists of 12 states and runs in parallel with the original finite state machine.

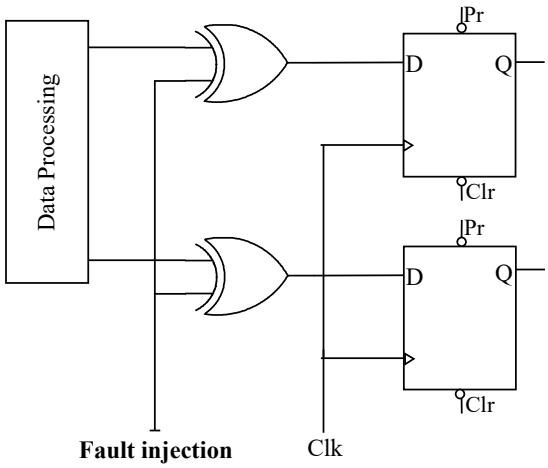

Fig. 5. Creation of a transient impulse.

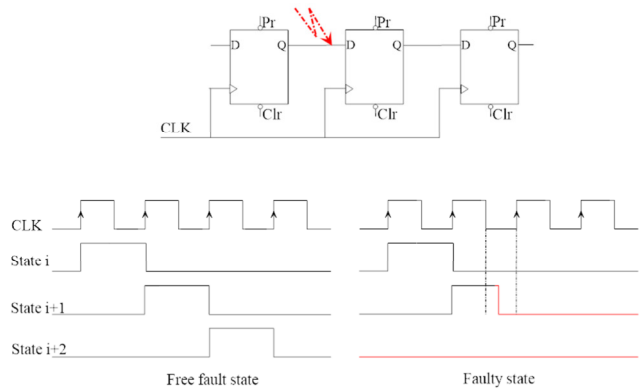

Fig. 6. Effect of injecting faults into the finite state machine.

IV. DETECTION CAPABILITY

The results from the two FSMs were compared at each clock cycle. If a difference is detected then both FSMs are reset (back to the initial state) and the various signals are set back to their initial values. Figure 7 illustrates the controller protection principle.

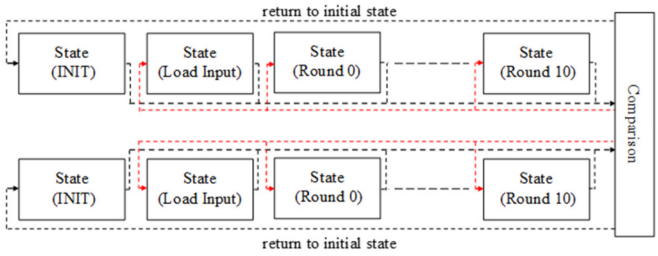

Fig. 7. Controller protection principle.

To ensure the robustness of the proposed system, a series of tests was carried out to evaluate its resistance against fault injection attacks. These tests involved injecting faults at various critical points within the system to assess its ability to withstand potential vulnerabilities:

- Injection of faults into the initial round.
- Injection of faults in the AES round.
- Injection of faults in the controller (disruption of the finite state machine and control signals).
- Injection of faults in various multiplexers and state registers.

The injected faults at the specified locations were meticulously analyzed. These transient faults, whether single or multiple, can have durations ranging from one to multiple clock cycles. Single transient faults affect only one data bit within a fault injection location. In contrast, multiple faults can affect multiple data bits across different locations that are randomly selected for injection. The analysis report generated by the proposed environment shows that errors in the hardware implementation of AES are partitioned into four classes.

- Silent faults are common in cryptographic systems, affecting specific transformations but not affecting the

overall result. They can be difficult to detect and may go unnoticed unless specifically tested for.

- Positive faults are a common occurrence during the encryption process, where errors may impact error detection flags but not the actual output of the encryption. Despite this, errors can still be identified within the system, highlighting the importance of thorough error detection mechanisms in place.
- Undetected errors can be particularly problematic as they may go unnoticed until they cause significant issues in the system. These errors can lead to corrupted data or incorrect results, posing a risk to the integrity and reliability of the encrypted information.
- Error detection is a process that identifies and rectifies errors in outputs, ensuring that any errors made can be easily pinpointed and rectified before they cause further issues, thereby preventing further problems from arising.

A. Simple Fault Attacks

The proposed system's parity-based error detection capabilities were evaluated under the assumption that attackers targeted only a single data bit. To assess the system's robustness against single fault attacks, 2,000,000 test vectors were generated using the proposed test environment to simulate various fault scenarios. Table I presents the experimental results on error detection capabilities, where the percentage of detected errors is calculated as the number of detected errors divided by the total number of injected faults, multiplied by 100 to obtain the percentage.

TABLE I. PROPOSED SCHEME ERROR DETECTION CAPABILITY

Fault attacks	Detection capability (%)			
	Positive Faults	Silent Faults	Undetected errors	Detected errors
Simple faults	16.56	15.59	0	67.85
Multiple faults (Random)	15.28	13.32	0.001	71.399

According to Table I, the proposed system demonstrated a strong capability in detecting simple transient faults, with an accuracy rate of approximately 67.85%. It is important to note that the system exhibits a negligible percentage of undetected faults, nearly 0%, with the remaining faults classified as either silent faults that do not affect system operation or false positives that indicate erroneous fault identifications.

B. Multiple Fault Attacks

To ensure the security of the proposed system against attacks targeting multiple data bits simultaneously, it is important to consider various scenarios where faults can be injected. By intentionally introducing errors into the encryption process and error detection mechanisms, the effectiveness of the proposed system in withstanding such attacks can be evaluated. This rigorous testing can help identify vulnerabilities and make the necessary improvements to improve the overall robustness of the proposed solution. The error detection system was extensively tested using 2,000,000 test vectors in a secure verification setting to assess its resilience against different fault attacks. Random bits from the 128 data bits were chosen to mimic the effects of multiple faults, and the results in Table I showcase the efficacy of the system in thwarting these attacks. The proposed system was highly effective in detecting faults during the attack process, with a probability of approximately 71.399%. However, there is a small chance (0.001%) that injected faults may go undetected, particularly if they target specific components, such as linear transformations, the multiplexer, or the initial round. This underscores the importance of ensuring that the modulo 2 addition between these components results in a non-zero value in each column of the state matrix to prevent undetected faults. A successful error detection system must reduce the likelihood of undetected errors while also avoiding false positives. Additionally, the system was experimentally proven to offer adequate protection against both natural and malicious fault attacks, ensuring a high level of security for users.

C. Fault Attacks on Critical Points

The proposed system's error detection capabilities, which are parity-based, were evaluated under the assumption that attackers target all the critical points of the AES algorithm, including the initial round, SubBytes and state register, multiplexer and linear transformations, and the controller. The experimental results on the error detection capabilities are presented in Table II.

TABLE II. ERROR DETECTION CAPABILITY: FAULT ATTACKS ON CRITICAL POINTS

Critical points	DETECTION CAPABILITY (%)				
	Injecting faults number	Positive Faults	Silent Faults	Undetected Errors	Detected Errors
Initial round	200000	13.646	12.843	0.0005	73.51
SubBytes and state register	800000	12.84	10.78	0.001	76.379
Multiplexer and linear transformations	800000	15.8	13.515	0.00125	70.68375
Controller	200000	13.83	13.85	0.001	72.32

This table presents the error detection capabilities of different critical points in the system, such as the Initial round, SubBytes and state register, multiplexer and linear transformations, and the Controller. The results show varying percentages of detected errors for each stage, with SubBytes and the state register having the highest detection capability at 76.379%. These findings highlight the importance of evaluating error detection mechanisms at different stages of the system to ensure robustness against faults.

V. HARDWARE IMPLEMENTATION OF FAULT DETECTION SCHEME

The VHDL model of AES was implemented on the Virtex5 FPGA platform from Xilinx with and without an error detection system. The hardware implementation was simulated using ModelSim 6.6 and synthesized with Xilinx ISE 14.6. Table III displays the results of the implementation, including the number of slices used, operating frequency in MHz, throughput in Mbps, as well as any additional area overhead

and degradation in frequency and data throughput for both scenarios.

TABLE III. IMPLEMENTATION OF THE PROPOSED DETECTION SYSTEM: RESULTS (DECREASE IS DENOTED BY USING THE '-' SIGN)

AES Design	Flip-Flops	Area (Slices)	Freq. (MHz)	Throu. (Mbps)
Unprotected AES	273	430	249.15	2657.6
Protected AES	392 (43.59%)	618 (43.72%)	239.60 (-3.83%)	2555.73 (-3.83%)

The hardware implementation of AES without error detection occupies an area of 430 slices at a frequency of 249.15 MHz. Protecting this implementation against fault attacks with the proposed error detection system results in an additional area cost of approximately 43.72% and a decrease in operating frequency of around 3.83% compared to the non-secure implementation. The area overhead cost of the secure implementation is primarily attributed to the incorporation of a second FSM in the controller and error detection systems to secure the four AES transformations, the initial round, and the state registers. This results in a significant increase in the number of flip-flops by approximately 43.59% compared to the original implementation, as shown in Table III.

Table IV presents a comparison between the proposed method and recently published works. It is worth noting that [11]^{a} and [11]^{b} propose error detection systems that only secure the four transformations of AES.

TABLE IV. IMPLEMENTATION OF THE PROPOSED DETECTION SYSTEM: COMPARISON (DECREASE IS DENOTED BY USING THE '-' SIGN)

Fault detection scheme	Fault Coverage (%)		Overhead (%)		
	Simple fault	Multiple fault	Area	Freq.	Throu.
[11]^{a}	100	99.9998	51	-21.24	-70.4
[11]^{b}	100	99.9927	36.7	-18.68	-69.5
Proposed	100	99.999	45.59	-3.83	-3.83

Although the proposed system serves to secure the entire hardware implementation of AES, it incurs a lower cost increase than that achieved by [11]^{a} and a lesser frequency degradation than that achieved by [11]^{b}. The detection system presented in [11]^{b} requires several modifications to the AES architecture, resulting in a data throughput degradation of up to 18 times compared to the proposed system. From a security standpoint, the proposed detection system results in an error detection capability of approximately 99.999%, making it extremely reliable for protecting the AES hardware implementation against attacks. Furthermore, this approach ensures enhanced security without significantly compromising overall system performance.

VI. SAFETY OF THE FAULT DETECTION FOR AES IN EMBEDDED SYSTEMS

The proposed fault detection scheme for AES in embedded systems demonstrates robust safety, achieving 99.999% fault coverage across all AES transformations, the initial round, and state registers, as validated by extensive simulations. This comprehensive protection effectively detects single- and multiple-bit errors, mitigating fault injection attacks. A comparison in Table IV highlights the scheme's superior fault coverage compared to existing methods ([11]^{a} and [11]^{b}) while maintaining a reasonable area overhead (45.59%) and minimal frequency degradation (3.83%). Although the added FSM and error detection systems contribute to area overhead, the impact on performance remains low, ensuring practicality for real-world deployments. This thorough approach, combined with empirical results, strongly supports the scheme's safety and reliability in protecting AES against fault attacks.

VII. CONCLUSION

This study presented a comprehensive analysis of a fault detection scheme for the AES algorithm. The results from both simulation and FPGA implementation showed that the proposed scheme is effective in detecting faults in the AES algorithm (approximately 99.999%). The experimental synthesis results demonstrated that the scheme can be efficiently implemented on FPGA platforms, with competitive performance in terms of area, frequency, and throughput. Overall, this research contributes to the field of fault detection in cryptographic algorithms and provides valuable insights for future research in this area. The successful implementation of this fault detection scheme not only enhances the security of the AES algorithm but also contributes to the advancement of sustainable development goals, particularly SDG 9 and SDG 16, by ensuring the integrity of cryptographic systems. This research paves the way for more secure and reliable communication systems in support of global efforts towards sustainable development.

ACKNOWLEDGMENT

The authors extend their appreciation to Prince Sattam bin Abdulaziz University for funding this research work through the project number (PSAU/2024/01/31267)

REFERENCES

[1] H. Mestiri, I. Barraj, A. Alsir Mohamed, and M. Machhout, "An Efficient AES 32-Bit Architecture Resistant to Fault Attacks," *Computers, Materials & Continua*, vol. 70, no. 2, pp. 3667–3683, 2022, https://doi.org/10.32604/cmc.2022.020716.

[2] T. M. Kumar, K. S. Reddy, S. Rinaldi, B. D. Parameshachari, and K. Arunachalam, "A Low Area High Speed FPGA Implementation of AES Architecture for Cryptography Application," *Electronics*, vol. 10, no. 16, Jan. 2021, Art. no. 2023, https://doi.org/10.3390/electronics10162023.

[3] S. S. S. Priya, P. Karthigaikumar, and N. R. Teja, "FPGA implementation of AES algorithm for high speed applications," *Analog Integrated Circuits and Signal Processing*, vol. 112, no. 1, pp. 115–125, Jul. 2022, https://doi.org/10.1007/s10470-021-01959-z.

[4] J. Breier and X. Hou, "How Practical Are Fault Injection Attacks, Really?," *IEEE Access*, vol. 10, pp. 113122–113130, 2022, https://doi.org/10.1109/ACCESS.2022.3217212.

[5] NCC Group, "An Introduction to Fault Injection (Part 1/3)," *NCC Group*, Jul. 07, 2021. https://www.nccgroup.com/sg/research-blog/an-introduction-to-fault-injection-part-13.

[6] M. R. Muttaki, M. H. Rahman, A. Kulkarni, M. Tehranipoor, and F. Farahmandi, "FTC: A Universal Framework for Fault-Injection Attack Detection and Prevention," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 32, no. 7, pp. 1311–1324, Jul. 2024, https://doi.org/10.1109/TVLSI.2024.3384531.

[7] A. Gangolli, Q. H. Mahmoud, and A. Azim, "A Systematic Review of Fault Injection Attacks on IoT Systems," *Electronics*, vol. 11, no. 13, Jan. 2022, Art. no. 2023, https://doi.org/10.3390/electronics11132023.

[8] S. Maiti and D. R. Chowdhury, "Design of fault-resilient S-boxes for AES-like block ciphers," *Cryptography and Communications*, vol. 13, no. 1, pp. 71–100, Jan. 2021, https://doi.org/10.1007/s12095-020-00452-0.

[9] M. Bedoui, H. Mestiri, B. Bouallegue, B. Hamdi, and M. Machhout, "An improvement of both security and reliability for AES implementations," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 10, pp. 9844–9851, Nov. 2022, https://doi.org/10.1016/j.jksuci.2021.12.012.

[10] S. Sheikhpour, S. B. Ko, and A. Mahani, "A low cost fault-attack resilient AES for IoT applications," *Microelectronics Reliability*, vol. 123, Aug. 2021, Art. no. 114202, https://doi.org/10.1016/j.microrel.2021.114202.

[11] S. Sheikhpour, A. Mahani, and N. Bagheri, "Practical fault resilient hardware implementations of AES," *IET Circuits, Devices & Systems*, vol. 13, no. 5, pp. 596–606, 2019, https://doi.org/10.1049/iet-cds.2018.5235.

[12] F. E. Potestad-Ordóñez, E. Tena-Sánchez, A. J. Acosta-Jiménez, C. J. Jiménez-Fernández, and R. Chaves, "Hardware Countermeasures Benchmarking against Fault Attacks," *Applied Sciences*, vol. 12, no. 5, Jan. 2022, Art. no. 2443, https://doi.org/10.3390/app12052443.