International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 6 June 2023, pp: 595-600 www.ijaem.net ISSN: 2395-5252

Detection of Malware Probing On FPGA

Prassanna Ganesan[1], SreeMurari K[2], Syed Fardeen Althaf[3], Mr. M. Athappan
123Department of Electronics and Communication Engineering, Sri Venkateswara College of Engineering, Sriperumbudur, Tamil Nadu

Date of Submission: 10-06-2023 Date of Acceptance: 20-06-2023

ABSTRACT: The semiconductor industry is currently encountering fresh challenges due to the unauthorized copying, reverse engineering, and extraction of critical design secrets during the life cycle of hardware intellectual property (IP). To counter these threats, a promising method called logic locking has emerged. It involves incorporating additional logic controlled by a secret key into strategic locations of an IP, effectively locking its functionality unless the correct key is available. This technique provides protection against IP piracy and misuse.

To evaluate the security of logic locking methods in existing systems, a novel attack known as synthesis-based constant propagation has been introduced. This attack focuses on analyzing each key-input port through synthesis-based techniques, searching for relevant design features that could potentially aid in determining the correct key value. The primary area of assessment lies in the built-in prediction model of the proposed system. This system comprises a machine learning algorithm that aims to detect attack occurrences as quickly as possible. It utilizes various attack constraints as attributes to make predictions. To achieve this, the system operates in two modes. The first mode is fully automated and continuously runs on-chip until it becomes active. The second mode involves a sleep-mode accelerator, which operates when the chip is in a static state. This accelerator extracts potential probing attacks that the integrated circuit may face.

I. INTRODUCTION

The continuous advancement of technology has led to the integration of FPGA and SoC architectures in numerous electronic systems. FPGAs offer the ability to dynamically configure hardware circuits, making them attractive for various applications. However, this flexibility also exposes these systems to potential security threats, including malware attacks. Traditional software-based security mechanisms are often insufficient to safeguard FPGA-based systems due to the complex and diverse nature of hardware configurations.

The objective of this paper is to present an effective solution for detecting malware probing activities on FPGA-based systems with SoC. Malware probing refers to the unauthorized attempts made by malicious software to gather information, exploit vulnerabilities, or compromise the integrity and confidentiality of the system. By identifying and mitigating such probing activities, system administrators can enhance the overall security of FPGA-based systems.

To achieve this goal, the proposed approach combines both hardware and software-based techniques. The hardware component involves the development of specialized monitoring circuits embedded within the FPGA fabric. These circuits are designed to monitor various system-level parameters, such as power consumption, clock frequency, and memory access patterns, which can indicate potential malware probing activities.

The software component of the proposed approach involves the implementation of intelligent algorithms that analyses the monitored data in real-time. These algorithms leverage machine learning and anomaly detection techniques to identify abnormal patterns and distinguish them from legitimate system operations. By continuously monitoring and analysing system behavior, the proposed approach can detect and raise alerts for potential malware probing activities, enabling timely response and mitigation.

The contributions of this paper include the design and implementation of the monitoring circuits, the development of intelligent algorithms for real-time analysis, and the evaluation of the proposed approach through comprehensive experiments and case studies. The results demonstrate the effectiveness of the approach in detecting various types of malware probing activities, thereby enhancing the security of FPGA-based systems with SoC.

II. LITERATURE REVIEW

Varga B., Kramoliš J., Kramolišová Z. (2018)"Malware Detection on FPGA Using Machine Learning Techniques." In this paper, the authors propose a machine learning-based approach for detecting malware on FPGAs. They explore the use of various machine learning algorithms and evaluate their effectiveness in detecting different types of malware.

Ranganathan N., Goettel B., Kastner R. (2014)"A Survey of FPGA-Based Side-Channel Attack and Countermeasure Techniques." This survey paper provides an overview of side-channel attack and countermeasure techniques implemented on FPGAs. It discusses different types of side-channel attacks and explores countermeasures to enhance the security of FPGA-based systems.

Shin M.K., Kim Y., Gu G., Xu D., Song D. (2015) "Malware in Firmware: Where It Hides and How to Find It." The authors investigate the presence of malware in firmware and propose techniques for its detection. While the paper focuses on firmware, the findings and detection methods can be adapted to FPGA-based systems with SoC.

Jaramillo F., Nahar J., Gebregiorgis A., Forte D. (2019)b"Detection of Hardware Trojan Attacks in FPGA Designs Using Machine Learning." This paper explores the use of machine learning techniques for detecting hardware Trojan attacks in FPGA designs. The authors propose a framework that leverages machine learning algorithms to identify and classify hardware Trojan activities.

Pramod A.P., Akhtar M., Malik F.H., Baharudin M.Z.B., Shaikh F.K. (2019) "A Survey on FPGA-Based Hardware Trojan Detection Techniques." The authors present a comprehensive survey of FPGA-based hardware Trojan detection techniques. They discuss different approaches, including side-channel analysis, behavioral monitoring, and anomaly detection, highlighting their strengths and limitations.

Yang J., Chakrabarty K. (2016)"Hardware Security Testing of Field-Programmable Gate Arrays (FPGAs)." This paper focuses on hardware security testing of FPGAs and discusses techniques for identifying vulnerabilities and security threats. It covers methods such as fault injection, reverse engineering, and side-channel attacks.

III. PROPOSED SYSTEM

3.1.OBJECTIVE

The aim of this research paper report is to propose an innovative approach for the detection of malware probing activities on FPGA-based systems integrated with System-on-Chip (SoC) architectures. The specific objectives encompass the development of a new method that synergistically combines hardware and software techniques to enhance system security. This involves the creation and integration of specialized monitoring circuits within the FPGA fabric to capture system-level parameters indicative of potential malware probing, including power consumption, clock frequency, and memory access patterns. Furthermore, intelligent algorithms will be designed to analyze the collected data in real-time, leveraging machine learning and anomaly detection methodologies to identify abnormal patterns associated with malware probing activities. To evaluate the effectiveness of the proposed approach, a comprehensive set of experiments and case studies will be conducted, assessing its capability to detect various types of malware probing activities and providing valuable insights into its performance and accuracy.

The ultimate objective is to fortify the security of FPGA-based systems by promptly detecting and generating alerts for potential malware probing, facilitating timely response and mitigation strategies. Additionally, this research paper report seeks to contribute to the existing body of knowledge in the domains of malware detection, FPGA-based systems, and security mechanisms by offering novel insights into the challenges and solutions pertaining to the detection and prevention of malware probing on FPGA-based systems with SoC integration.

3.2.SYSTEM MODEL:

Logic locking is a promising technique for safeguarding hardware intellectual property (IP) from attacks. It involves the insertion of additional logic, controlled by a secret key, at strategic points within the IP to render its functionality inaccessible without the correct key. To evaluate the effectiveness of logic locking methods in existing systems, a novel synthesis-based constant propagation attack is proposed. This attack focuses on analyzing each key-input port through synthesis-based analysis to identify design characteristics that could potentially aid in deducing the correct key value.

The evaluation primarily focuses on the performance of the prediction model embedded within the proposed system. The system employs a machine learning algorithm that aims to swiftly identify the presence of an attack by considering various attack constraints as input attributes. To achieve this, the system operates in two modes, with the fully automated mode continuously running on-chip until activation occurs. The algorithm effectively identifies patterns of abnormality within the provided data input. The system is capable of detecting different types of attacks, including Forbidden attacks, Baiting attacks, Corruption attacksand Side Channel attacks. During the evaluation, physical attacks are injected externally, and the simulation accurately classifies the specific type of attack encountered.

Forbidden attack:

A forbidden attack can occur when an attacker tries to access a restricted area of the IC or modify its configuration to gain unauthorized access to sensitive data or resources. FPGAs are reconfigurable integrated circuits that allow users to customize and implement digital logic circuits. These attacks exploit vulnerabilities or weaknesses in FPGA devices to gain unauthorized access, manipulate functionality, or compromise their security.

Baiting attack:

A baiting attack is a type of cyberattack where an attacker uses a lure or bait, such as a fake USB drive to entice a user into plugging it into an SoC-based device. The attacker may use a baiting attack to gain unauthorized access to sensitive data, install malware or carry out other nefarious activities.

Corruption attack:

A corruption attack is a type of attack where an attacker attempts to alter or damage data, systems, or applications in order to disrupt their proper functioning. Corruption Attacks on SoC can have severe consequences, including unauthorized access, data loss, system instability, or compromise of sensitive information.

Side Channel attack:

A side-channel attack on a is a type of attack that targets the unintended information that leaks from a system during its normal operation, rather than exploiting vulnerabilities in the system's software or hardware directly.

These attacks are based on analyzing the physical properties or environmental factors of the device, such as power consumption, electromagnetic radiation, or sound, that leak information about the device's internal state. During side channel attacks, there is also a possibility of the occurrence of glitches.

3.2.1 MODELSIM SIMULATOR

Digital system designers are inevitably confronted with the critical task of testing their designs. Each design consists of multiple components, each requiring individual testing before integration into the overall design. Simulation is the primary method used to verify the correct operation of a design. It involves applying inputs to a circuit and observing its behavior, generating a set of waveforms that depict the circuit's response based on the given inputs.

Two main types of simulation are commonly used: functional simulation and timing simulation. Functional simulation focuses on testing the logical functionality of a circuit without considering delays within the circuit. Here, signals propagate through the circuit assuming zero logic and wiring delays. It is fast and serves as a valuable tool for verifying the fundamental correctness of the circuit design.

On the other hand, timing simulation is a more complex form of simulation that accounts for the time taken by logic components and wires to respond to input stimuli. In addition to testing the logical operation of the circuit, timing simulation provides insights into the timing behavior of signals within the circuit. This type of simulation is more realistic than functional simulation but requires a longer duration to perform due to the consideration of delays.

By employing a combination of functional and timing simulations, designers can comprehensively test and validate their digital designs, ensuring both logical correctness and appropriate timing behavior of the circuits.

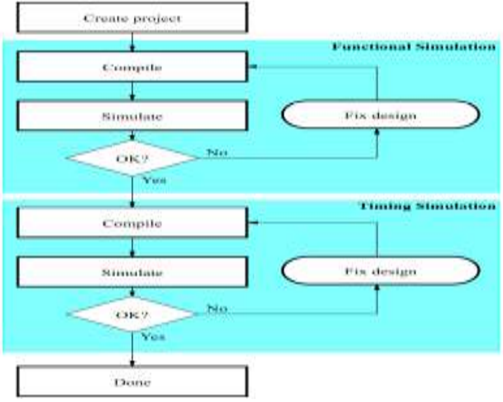

Fig 1. Simulation Flow

3.2.2 VHDL

VHDL, which stands for VHSIC Hardware Description Language, derives its acronym from VHSIC (Very High-Speed Integrated Circuits). The language is designed to capture the essence of hardware description, similar to how schematics are used. While VHDL can be perceived as complex, its acronym serves the purpose of reflecting its focus on hardware description.

VHDL serves multiple purposes, including documentation, verification, and synthesis of large digital designs. One of its key advantages is that the same VHDL code can be used for all three purposes, saving significant effort. In describing hardware, VHDL offers three main methodologies: structural, data flow, and behavioral approaches. Typically, a combination of these methodologies is employed depending on the specific requirements. The subsequent sections delve into each of these methodologies, providing an introduction to VHDL's usage in different contexts. Additionally, there are guidelines and recommended practices for utilizing VHDL for synthesis.

VHDL has become an industry standard, recognized by both the IEEE and the U.S. Department of Defense, for describing electronic system designs. As experience with VHDL grows and supporting tools become more accessible, it is gaining popularity in private industry as well. Consequently, a solid understanding of VHDL is increasingly important to facilitate the exchange of system description information in various domains.

3.2.3 FPGA

Introduced in 1984, field-programmable gate arrays (FPGAs) emerged as a viable alternative to programmable logic devices (PLDs) and application-specific integrated circuits (ASICs). FPGAs offer the distinct advantage of being programmable, allowing designers to modify their circuits multiple times. This flexibility has led to innovative designs utilizing FPGAs as a platform for implementation. However, there are considerations to be aware of when working with FPGAs. The economics of FPGAs require designers to strike a balance between their relatively higher piece-part pricing compared to ASICs and the absence of high non-recurring engineering (NRE) costs and lengthy development cycles. Additionally, FPGAs are available in fixed sizes, making efficient utilization of silicon area a significant factor.

FPGAs bridge the gap between discrete logic and smaller PLDs on the lower end of the complexity scale and expensive custom ASICs on the higher end. They consist of an array of configurable logic blocks connected through programmable interconnects. Surrounding the logic blocks are programmable I/O blocks. The programming technology used in an FPGA determines the type of logic cells and interconnect scheme employed, which in turn dictate the design of input/output circuits and the programming method.

FPGAs provide a comprehensive set of features required to implement complex designs. On-chip phase-locked loops (PLLs) or delay-locked loops (DLLs) facilitate clock management. Dedicated memory blocks can be configured as single-port RAMs, ROMs, FIFOs, or CAMs. The logic fabric within the FPGA enables diverse data processing capabilities. FPGAs also support various single-ended and differential I/O standards, facilitating integration with backplanes, high-speed buses, and memories. Additional system-building resources, such as high-speed serial I/Os, arithmetic modules, embedded processors, and ample memory, are commonly found in modern FPGAs.

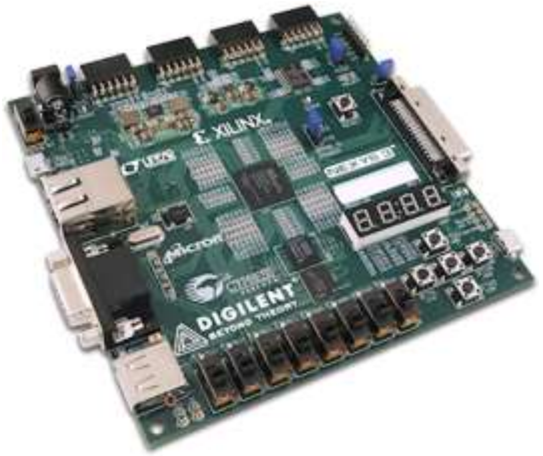

Fig 2. Nexys3 Spartan-6 FPGA Board

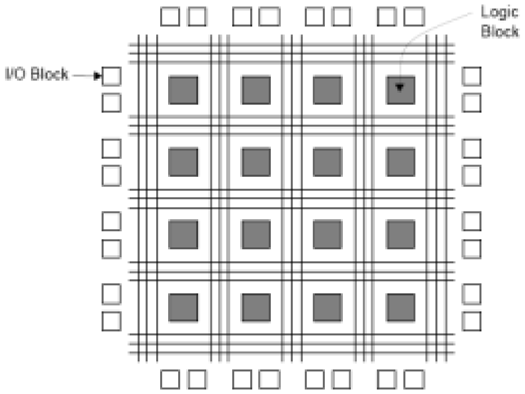

Fig 3. Illustration of a typical FPGA

The figure above illustrates a typical FPGA architecture. With their programmability and ability to support high logic capacity, FPGAs have fundamentally transformed the way digital circuits are designed and implemented.

IV. MODULE DESCRIPTIONS

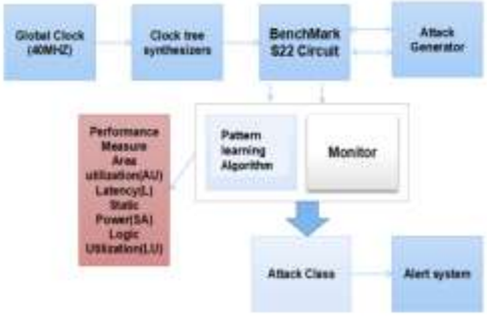
Global Clock (40MHZ)
Clock tree synthesizers
BenchMark S22 Circuit
Attack Generator
Performance Measure Area utilization(AU) Latency(L) Static Power(SA) Logic Utilization(LU)
Pattern learning Algorithm
Monitor
Attack Class
Alert system

Fig 4. Block Diagram

MODULE 1: DESIGN OF TEST CIRCUIT

The module includes a benchmark test circuit that serves as a crucial component for evaluating the proposed system. This test circuit allows the deployment of the application and facilitates the generation of uninterrupted outputs. Specifically, the test circuit incorporates a clock synthesizer that enables the functioning of the Benchmark S22 logic, ensuring the smooth operation of the overall system

MODULE2:DESIGN OF ATTACKSCENARIO GENERATOR

This focuses on detecting various types of attacks, including forbidden attacks, corruption attacks, baiting attacks and side channel attacks. These attacks are generated and simulated to disrupt the normal behavior of the SOC, thereby providing a comprehensive testing process. By cloning and simulating these attacks, the system model enables the identification and mitigation of abnormal activities, ensuring the integrity and security of the FPGA-based SOC.

MODULE 3: DESIGN OF ALGORITHM

The module incorporates machine learning techniques using machine learning algorithms, which are optimized based on the clock frequency. The aim is to develop a model that can detect attack patterns after being trained. The training dataset constitutes 80% of the data, while the remaining 20% is reserved for testing purposes. The attack patterns are classified by analyzing the occurrence of clock errors originating from the test circuit.

V. SIMULATION RESULTS:

Fig 5. Waveform of Forbidden Attack

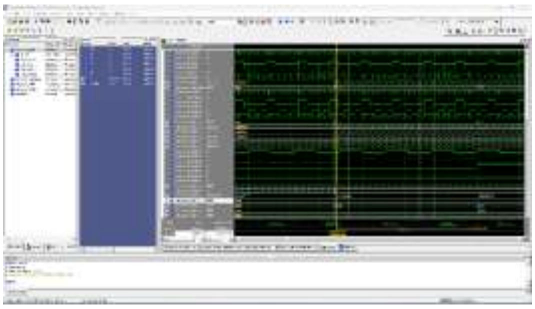

Fig 6. Waveform of Corruption Attack

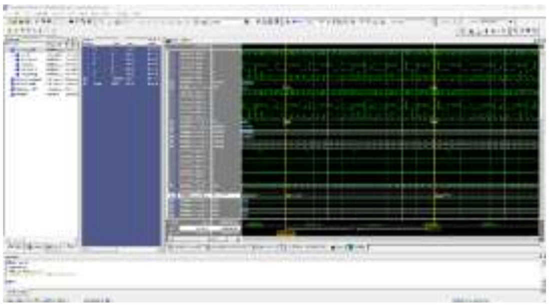

Fig 7. Waveform of Baiting Attack

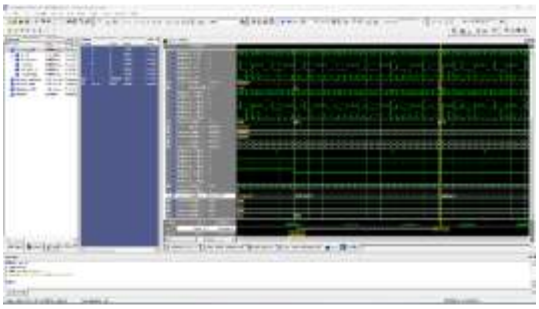

Fig 8. Side Channel

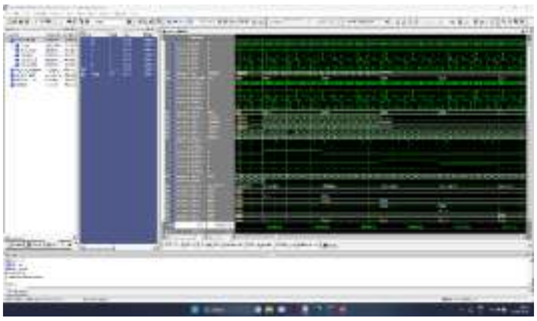

Fig 9. Full Waveform

VI. CONCLUSION & FUTURE SCOPE:

Conclusion: The project employed various techniques and algorithms to analyze the behavior of malware probing attacks specifically targeting FPGA-based SoCs. The objective was to leverage the flexibility and programmability of FPGAs to create a dynamic detection system capable of adapting to evolving malware threats.

The developed detection system showcased promising results in detecting and identifying malware probing attacks. By monitoring and analyzing the system's behavior at different levels, encompassing both hardware and software, potential instances of malware probing activities were successfully detected and flagged for further investigation or mitigation.

This research contributes to enhancing the security of digital circuits and SoC devices by providing an effective means of detecting and responding to malware probing attacks. Further advancements in this area could lead to improved security measures and better protection against evolving cyber threats in the digital domain.

Future Scope: While the project has successfully accomplished its primary objectives, there are still opportunities for further research and development in the field of detecting malware probing on FPGA-based SoCs.

Future research efforts could be devoted to refining the detection techniques employed in the project. Exploring machine learning and artificial intelligence approaches may lead to the development of more advanced algorithms capable of detecting sophisticated malware probing techniques.

Although the project primarily focused on detection, future work could concentrate on the development of real-time response mechanisms. This might involve implementing automated countermeasures or proactive measures to effectively neutralize identified malware probing attempts.

The project has established a strong foundation for the detection of malware probing on FPGA-based SoCs. Additional research and development in the suggested future directions have the potential to make significant contributions to the security of FPGA-based systems and help address the evolving threats posed by malware probing attacks.

REFERENCES

[1]. Ayush Jain, Ziqi Zhou and Ujjwal Guin, “TAAL: Tampering Attack on Any Key-based Logic Locked Circuits,” ACM Transactions on Design Automation of Electronic Systems, Vol. 26, Issue 4, pp. 1–22, Mar. 2021.

[2]. M. Tanjidur Rahman, M. Sazadur Rahman, Huanyu Wang, Shahin Tajik, Waleed Khalil, FarimahFarahmandi, Domenic Forte, NavidAsadizanjani and Mark Tehranipoor, “Defense-in-depth: A recipe for logic locking to prevail,” Integration, the VLSI Journal, Vol. 72, Issue. C, May 2020, pp. 39–57.

[3]. Deepak Sirone and Pramod Subramanyan, “Functional Analysis Attacks on Logic Locking,” IEEE Transactions on Information Forensics and Security, Vol. 15, Jan. 2020, pp. 2514–2527.

[4]. Kumar, A., Mahadevan, K., Joseph, A., & Raha, S. (2015). High-performance malware detection using hardware-based virtualization.IEEE Transactions on Dependable and Secure Computing, Vol. 12, Issue 4, pp. 377-390.

[5]. López, M., Ribagorda, A., & Sánchez-Artigas, M. (2014). Hardware-based malware detection using artificial immune system and associative memories. Computers & Security, Vol. 43, pp. 111-122.