iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** VII **Month of publication:** July 2026

DOI: https://doi.org/10.22214/ijraset.2026.84348

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VII July 2026- Available at www.ijraset.com

FPGA Camouflage Detection for SOC Devices

Ancy R[1], Jebisha Sherlin Anne J[2], Logeshwari D[3], Dr. G. Balachandran[4]
[1, 2, 3]*Bachelor of Engineering in Electronics and Communication Engineering, Jeppiaar Engineering College, Chennai, ANNA University, CHENNAI*
[4]*M.Tech., Ph.D., Associate Professor Department Of Electronics And Communiaction Engineering, Jeppiaar Engineering College, Chennai*

Abstract: ***The growth of SOC devices in the latest technologies going high every day. Powerful side-channel analysis (SCA) attacks based on failure analysis (FA) techniques can bypass conventional countermeasures on integrated circuits (ICs), and therefore, break the entire system's security. Laser Logic State Imaging (LLSI) from the IC backside is an example of such attacks, making the contactless probing of static on-die signals possible. Several countermeasures have been proposed to prevent optical probing attacks, such as LSI. However, these schemes are designed according to the laser properties and its impact on transistors, and hence, they have complex fabrication steps and large area overhead. As a result, they are difficult to verify and implement. In this paper, we propose a twofold detection self-timed sensor, which is the first attempt, to our knowledge, for an easy-to implement circuit-based countermeasure to thwart LLSI attacks. To perform LLSI, the attacker needs to freeze the clock at a point of interest and modulate the voltage supply line at a known frequency to leak the state of transistors through laser light reflections. With these two attack requirements in mind, we design, simulate, and implement clock- and voltage-based sensors that can detect VLSI attacks with very high confidence.***

I. INTRODUCTION

A. Background

FPGA-based systems are widely used in modern electronics due to their flexibility and reconfigurable nature. The involvement of vendors, foundries, and system developers introduces security challenges during design and manufacturing. Without proper protection, FPGA systems are vulnerable to threats like hardware Trojans and reverse engineering.

- Multiple parties involved in FPGA design and manufacturing
- Risk of hardware Trojan insertion during fabrication
- Possibility of reverse engineering of FPGA bitstreams
- Need for secure techniques to protect FPGA systems

With the increasing use of FPGA-based systems, there is a need to implement security techniques to protect designs. Methods such as obfuscation and logic locking help prevent unauthorized access. These approaches improve system security and reliability.

B. Problem Statement

FPGA-based systems involve multiple stakeholders, making them highly vulnerable to security threats during design, fabrication, and deployment stages. Attacks such as hardware Trojan insertion, reverse engineering, cloning, and replay attacks can compromise system integrity and confidentiality.

- In the existing system, only Physical attacks are considered for analysis
- Laser injection-based problems are discussed.
- Voltage modulation sensor, two-fold sensor validation is done.
- Need for secure methods to protect FPGA systems

With the increasing dependence on FPGA-based applications, it is essential to develop effective security mechanisms to prevent unauthorized access and ensure system protection.

C. Objectives

The primary objective of this project is to address security vulnerabilities in FPGA- based systems by applying logic locking and obfuscation techniques. The system focuses on protecting IP cores and preventing attacks such as reverse engineering, cloning, and hardware Trojan insertion.

- Implement logic locking mechanisms such as XOR/XNOR gates
- Apply obfuscation techniques like multiplexers and functional stripping
- Protect FPGA designs from reverse engineering and cloning
- Enhance security against hardware Trojans and replay attacks

The system also focuses on improving trust between different FPGA stakeholders by ensuring secure design and protection of sensitive information.

D. Scope Of The Project

The project focuses on analyzing the security and trust vulnerabilities in FPGA-based systems by considering different parties involved in the FPGA market model, including FPGA vendors, foundries, system developers, IP core vendors, EDA tool vendors and end users. It studies the interactions between these entities and identifies the security risks that arise during design, fabrication and deployment of FPGA systems.

- Identification of hardware Trojan threats in FPGA-based systems
- Evaluation of reverse engineering and cloning risks in FPGA designs
- Analysis of replay and side-channel attacks in FPGA configurations
- Implementation of logic locking techniques for improving system security
- Identification of information leakage risks in FPGA-based systems

The project is designed with a specific focus on FPGA-based systems, particularly those involving multiple stakeholders such as vendors, foundries, system developers, and end users. These system's face significant security challenges during design fabrication and deployment stages, including hardware Trojans, information leakage, reverse engineering and cloning, By implementing logic locking and obfuscation techniques, the project aims to provide a practical and scalable solution to these security threats. The system's ability to protect FPGA configurations and prevent unauthorized access will enhance the overall security of FPGA-based applications, ultimately helping to ensure reliable operation, safeguard sensitive data, and improve trust across different stages of system development.

II. LITERATURE REVIEW

A. Research Studies On Fpga Camouflage Detection

Several research studies have explored security challenges in FPGA-based systems focusing on threats such as hardware Trojans, reverse engineering, and cloning attack. These studies highlight the need for protection mechanisms during design fabrication, and deployment stages.

- Logic locking techniques using XOR/XNOR gates for security
- Obfuscation methods using multiplexers and functional stripping
- Key-gate insertion strategies for protecting FPGA designs
- Protection against reverse engineering and cloning attacks

Research indicates that implementing these techniques improves the security of FPGA systems, protects intellectual property, and reduces vulnerabilities across different stages of system development.

1) Title*:* Hardware Trojan Attacks on FPGA-Based CNN Accelerators and Detection Technique
Author: Jia Hou, Zic hu Liu, Zepeng Yang, Chen Yang Abstract:
This paper investigates technique using physically unclonable functions (PUFs) to identify and localize malicious hardware modifications. The system evaluates accelerator data paths and control logic for abnormal responses. Results show improved Trojan detection and protection with minimal impact on performance hardware Trojan attacks targeting FPGA-based CNN accelerators and their impact on inference accuracy. It proposes a detection
Year: 2024

2) Title*:* Hardware Trojan Detection Using Laser Logic State Imaging Author: Thilo Krachenfels, Jean-Pierre Seifert, Shahin Tajik Abstract: This paper presents a non-invasive technique to detect dormant hardware Trojans in FPGA systems using laser logic state imaging. The method captures internal logic states and compares them to trusted configurations. It identifies even small modifications in the FPGA fabric. Results show enhanced detection accuracy and improved hardware security.
Year: 2023

3) Title: A New Logic-Locking Scheme Resilient to Gate Removal Attack Authors: Jingbo Zhou; Xinmiao Zhang
Abstract:
This paper focuses on improving logic locking techniques to resist SAT and removal attacks in integrated circuits. Existing schemes depend on a single product term, making them vulnerable when identified and removed. The proposed method uses unique patterns to maintain security even after partial gate removal, improving resistance against SAT-based attacks.
Year: 2020

4) Title: A CRISPR-Inspired Mechanism for Detecting Hardware Trojans in FPGA Devices
Author: Dillon Staub et al Abstract:
This paper presents an FPGA-based camouflage detection system for SoC devices. It uses machine learning to identify abnormal circuit behaviors and potential hardware attacks. The system employs logic-locking to protect critical IP blocks and triggers real-time protective actions. Results show improved security and reliability with minimal performance overhead.
Year : 2020

5) Title: Dynamic FPGA Detection and Protection of Hardware Trojan Author: Amr Alanwar, Mona A. Aboelnaga, Yousra Alkabani, M. Watheq El-Kharashi, Hassan Bedour
Abstract:
This paper presents a runtime FPGA hardware Trojan detection and protection system that does not require a trusted reference design. It monitors internal circuit behavior to identify anomalous patterns caused by malicious modifications. The system activates mitigation mechanisms to isolate or remove harmful logic. Results demonstrate enhanced reliability and reduced security risks during FPGA operation.
Year:2017
CONCLUSION
The reviewed literature underscores the increasing threat of Hardware Trojans in FPGA-based systems and integrated circuits, with researchers proposing diverse countermeasures ranging from PUF-based detection and laser logic state imaging to machine learning-assisted identification and runtime monitoring.

III. METHODOLOGY

In the existing system, the scenario pertains to a physical attack targeting System-on-Chip (SOC) devices, where the attack's impact on these devices is assessed using voltage-based sensors within a VLSI simulation. This simulation aims to demonstrate the effectiveness of an attack detection mechanism in a hardware security context.

A. Proposed System

To overcome various clock issues in System on chip devices, multi-tasking FPGA chips need to be tested and validated with physical attack detection modules. here an in-depth analysis module is created to test the FPGA devices with multiple clock tree synthesizers such as sweep generators, configurable clock synthesizers, Digital phase locked loop etc.
Advantages of proposed system

- To explore the idea of physical attack detection in SOC devices, application circuits need to be developed and tested.
- Multi-tasking clock synthesizers, Clock distributors need to be tested with multiple resources

B. Implementation Summary

In your scenario, it seems that you are conducting a simulation to assess how voltage-based sensors can detect the impact of physical attacks on SOC devices. This simulation would involve modeling the SOC device, the voltage-based sensors, and various types of physical attacks to evaluate the effectiveness of the attack detector.
Some possible steps in such a simulation could include:

- Developing a detailed model of the SOC device, including its components and power consumption characteristics.
- Implementing voltage-based sensors to monitor the SOC device's voltage levels.
- Simulating different types of physical attacks, such as voltage spikes, signal interference, or attempts to tamper with the hardware.
- Analyzing the sensor data to detect anomalies or deviations from expected behavior that may indicate an attack.

C. Features, Innovation & Advantages

The proposed FPGA camouflage detection system stands out by integrating advanced techniques such as machine learning for real-time anomaly detection and logic-locking for IP protection. It also employs continuous monitoring of internal FPGA signals and low-power design strategies to maintain performance. These features work together to enhance security, reduce vulnerability to hardware Trojans, and protect critical IP blocks—making it a robust solution for modern SoC devices.

Key Features and Benefits:

- Machine learning enables accurate detection of abnormal circuit behavior
- Logic-locking secures critical IP blocks against reverse engineering
- Real-time monitoring triggers protective actions instantly
- Low-power design maintains FPGA efficiency and performance
- Reduces reliance on offline verification, saving time and cost

D. Modules & Their Description

The FPGA Camouflage Detection System is composed of several specialized modules, each performing a key function in detecting hardware attacks in SoC devices. These modules work together to generate test conditions, simulate attacks, monitor system behavior, and detect anomalies efficiently.

1) Reference Clock Generator Module

This module generates the required clock signals for the FPGA system using a global clock and reset input. It ensures proper synchronization and timing for all other modules in the system.

- Implements clock divider to generate required frequencies
- Provides stable reference timing for operations
- Ensures synchronized functioning of all modules
- Uses global clock and reset signals

2) Test Circuit Generation Module

This module creates test circuits to simulate real-time FPGA operating conditions and evaluate system performance under different scenarios.

- Includes sweep generator, clock synthesizer and digital PLL
- Generates multiple clock signals for testing

3) Attack Pattern Generation Module

This module generates different hardware attack patterns to test the security of the FPGA system against various threats.

- Simulates side-channel attack conditions
- Generates replay attack patterns
- Produces corruption-based attack signals
- Helps evaluate system vulnerability and robustness

4) Design Of Dynamic Pattern Matching Algorithm

The module contains the dynamic pattern matching algorithm evaluated for the purpose of detection and validation of side channel attacks, FPGA replay attacks, corruption attacks etc.

5) Integration

The fine state machine enabled process is designed and connected with the sub modules of the process, further the integration of all modules that work without any correlation. The integration module orchestrates seamless interaction between all sub-modules using a finite state machine that governs transitions between idle, monitoring, detection, and response phases. A synchronization layer aligns clock domains, preventing race conditions and ensuring signal integrity. Inter-module communication is standardized through defined interface protocols, and the complete system is mapped onto the FPGA fabric, achieving reliable end-to-end detection with minimal area and power overhead.

E. Design Description

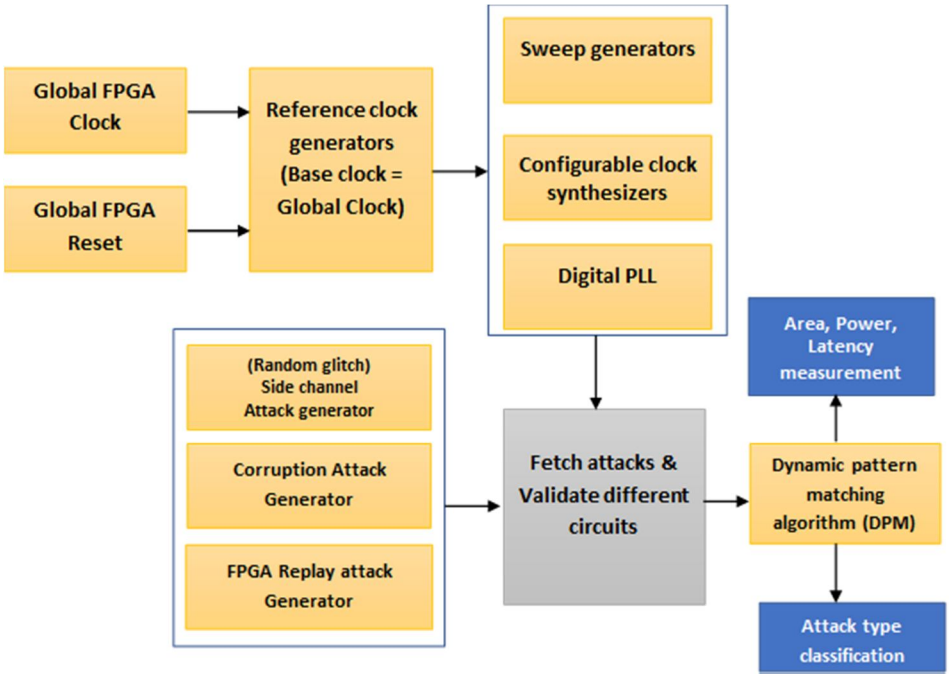


Fig 3.1 Design Description

Detailed Proposed

In addressing the intricate clock-related challenges prevalent in System-on-Chip (SoC) devices, a comprehensive testing and validation approach is employed, specifically focusing on multi-tasking FPGA chips. Recognizing the vulnerabilities associated with these devices, a key strategy involves the integration of physical attack detection modules to enhance security and resilience against potential threats.

A critical component of this testing framework is the in-depth analysis module, designed to thoroughly evaluate FPGA devices. This module employs various clock tree synthesizers, including sweep generators, configurable clock synthesizers, and Digital Phase-Locked Loops (DPLLs).

F. Hardware Descriptions

The Xilinx XC9572XL is part of the Cool Runner-II family of programmable logic devices (PLDs). It is a CPLD (Complex Programmable Logic Device) designed to offer a flexible, cost-effective solution for implementing custom logic in a variety of applications.

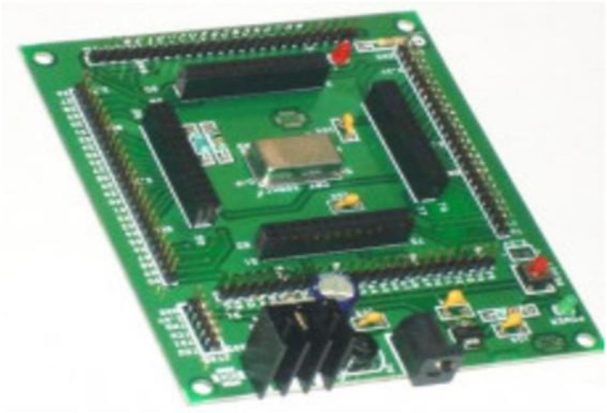


Fig 3.2 Xilinx XC9572XL

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VII July 2026- Available at www.ijraset.com

1) Programmable Logic Blocks (PLBS

Fig 3.3 Programmable Logic Blocks (Plbs)

2) Xilinx Flah Blast Programmer

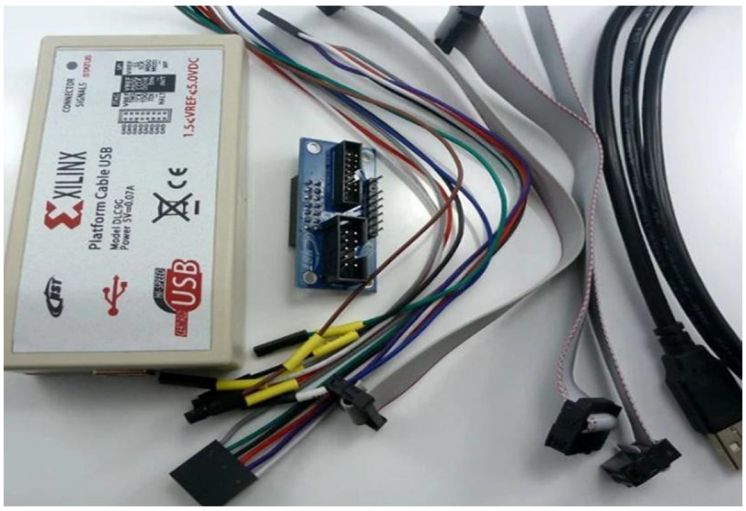


Fig 3.4 Xilinx Flah Blast Programmer

Xilinx JTAG Programming is a method used to configure and program Xilinx devices, including FPGAs (Field-Programmable Gate Arrays), CPLDs (Complex Programmable Logic Devices), and SoCs (System on Chips), through a standard interface known as JTAG (Joint Test Action Group). JTAG programming utilizes the IEEE 1149.1 standard and enables users to program devices by directly communicating with the internal registers and memory cells through a dedicated JTAG port.

3) Overview and Workflow

The process begins by connecting a JTAG programming tool (such as the Xilinx Platform Cable USB or the Digilent JTAG programmer) to the JTAG header on the Xilinx device.

G. Programming Procedure

Design and Synthesis: The user first designs their logic or circuit using a hardware description language (HDL) like VHDL or Verilog. Once the design is complete, it is synthesized, optimized, and mapped into a configuration bitstream using Xilinx's tools (Vivado, ISE).

Programming the Device: After synthesizing the design into a bitstream file, the file is sent to the target device via JTAG. The JTAG programmer sends the bitstream through a series of data registers and memory cells in the target device. These registers are part of the device's configuration logic, and the bitstream defines how the programmable logic should be set up.

Verification: After programming, the JTAG interface allows for a quick verification process to ensure that the bitstream was correctly loaded and that the device operates as expected. The JTAG tool can perform boundary scan tests, checking the integrity of the signals and the proper connection of pins.

Real-time Debugging: One of the significant advantages of JTAG programming is the ability to debug designs in real-time. Through the JTAG interface, engineers can monitor the internal state of the FPGA or CPLD during operation, making it easier to pinpoint issues such as timing violations, logic errors, or incorrect configurations.

H. Software: Modelsim 6.3g Altera

The FPGA Camouflage Detection System utilizes simulation software to verify and validate the functionality of hardware modules before implementation. ModelSim 6.3g (Altera Edition) is used as the primary simulation tool for designing and testing the Verilog-based system. The software runs on a Windows platform and provides an efficient environment for functional verification and debugging of digital circuits.

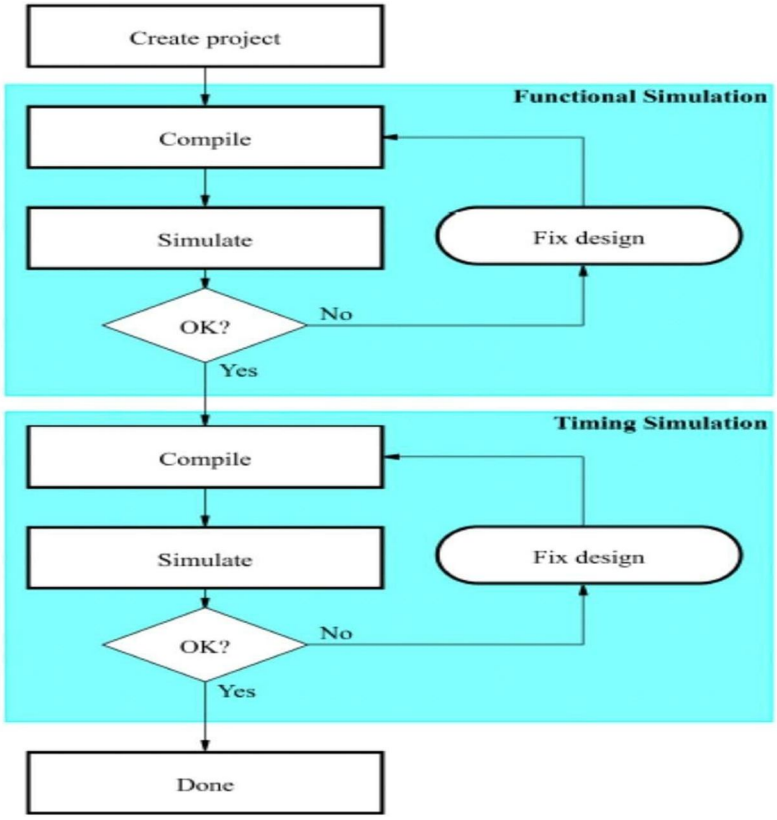
Software Stack:

- Operating System: Windows 7/10
- Hardware Description Language: Verilog HDL
- Simulation Tool: ModelSim 6.3g (Altera Edition)
- Design Entry: Verilog coding and testbench creation

I. Modelsim Simulator

Designers of digital systems are inevitably faced with the task of testing their designs. Each design can be composed of many components, each of which has to be tested in isolation and then integrated into a design when it operates correctly. To verify that a design operates correctly we use simulation, which is a process of testing the design by applying inputs to a circuit and observing its behavior. The output of a simulation is a set of waveforms that show how a circuit behaves based on a given sequence of inputs.

1) The Simulation Flow



ig 3.5 Simulation Flow

2) VHDL

VHDL is an acronym which stands for VHSIC Hardware Description Language. It is being used for documentation, verification, and synthesis of large digital designs. This is actually one of the key features of VHDL, since the same VHDL code can theoretically achieve all three of these goals, thus saving a lot of effort.

Design Flow

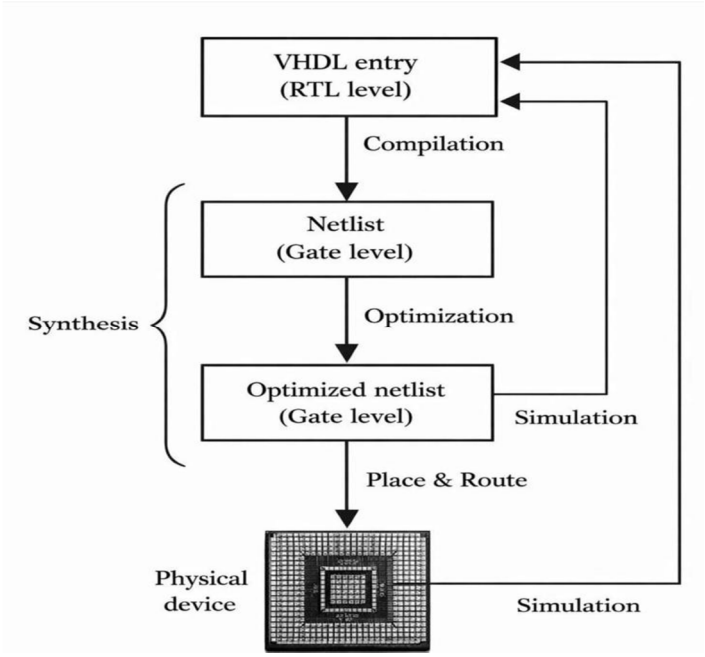


Fig 3.6 Design Flow

3) FPGA

The highest capacity general purpose logic chips available today are the traditional gate arrays sometimes referred to as Mask-Programmable Gate Arrays (MPGAs). MPGAs consist of an array of pre-fabricated transistors that can be customized into the user's logic circuit by connecting the transistors with custom wires. FPGA configuration is performed through programming by the end user. An illustration of a typical FPGA architecture appears below figure. As the only type of FPD that supports very high logic capacity, FPGAs have been responsible for a major shift in the way digital circuits are designed.

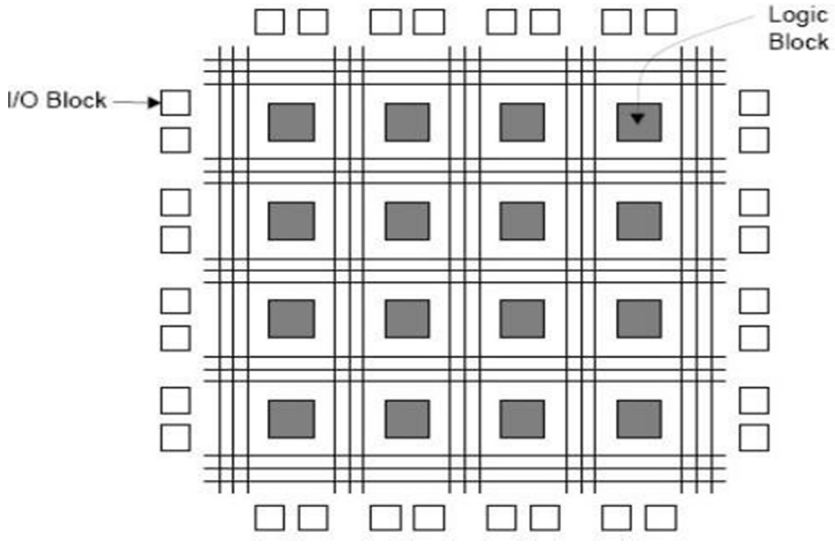


Fig 3.7 Fpga Architecture

J. Vulnerabilities and Attacks

We consider the following major parties in the FPGA-based systems market:

FPGA vendors: These are the companies (like Xilinx and Intel) that design FPGA architecture and build FPGA chips. These chips normally come with some built-in functional units, memory blocks, and other IP components that the FPGA vendors believe the customers will need.

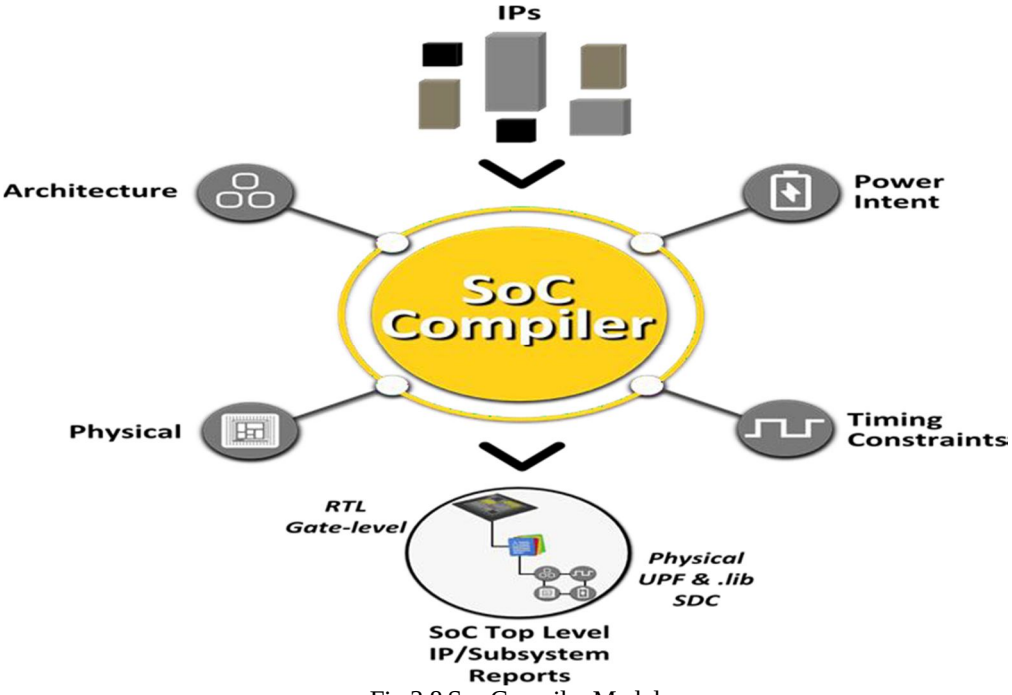


Fig 3.8 Soc Compiler Module

K. Security And Trust Vulnerabilities

In this section, we analyze the vulnerabilities at each interaction between a pair of parties in the above FPGA-based systems. The general guideline is that on the sending side of an arrow, the party that sends out the request will have concerns about the security of the information or product he sends out; on the receiving end of an arrow, the party needs to verify whether the received service or product is trusted.

FPGA vendors and foundries: (a) overbuilding: an untrusted foundry may fabricate more FPGA chips than required and sell them at a lower price to system developers. (b) hardware Trojan: during the fabrication process, unwanted functionalities, known as hardware Trojan, may be embedded in the FPGA chips; (c) information leaking: FPGA vendor's confidential data needed for the fabrication of the FPGA chips may be mishandled or leaked to parties (e.g. another competing FPGA vendor) that should not have access to such data.

FPGA vendors (or system developers) and IP core vendors (or EDA tool vendors): (b) hardware Trojan: FPGA vendors (or system developers) need to ensure the IPs provided by IP vendors do not contain malicious Trojans; (c) information leaking: malicious programs in EDA tools may take advantage of the Xilinx FPGA design suite's backdoor to manually disturb the FPGA design after placement and routing or collect valuable data about FPGA chip and/or the system; (d) IP protection: IP core vendors and EDA tool vendors want that their IPs and tools are used and royalty are paid properly. (e) reverse engineering: IP core vendors expect their IPs cannot be reverse engineered by untrusted parties.

FPGA-based system developers: these are the companies that create commercial products on FPGA chips. The product is normally in the form of configuration bit stream file for a given family FPGA chips.

FPGA-based system end users: these are companies or individuals who obtain the FPGA-based systems (like network routers, TV set-top boxes) from the system developer and use these systems.

IP core vendors: these are the companies that develop IP cores (like memory blocks, DSP cores) for specific applications, not necessary for FPGA-based systems.

IV. RESULTS AND DISCUSSIONS

A. *Execution Results*

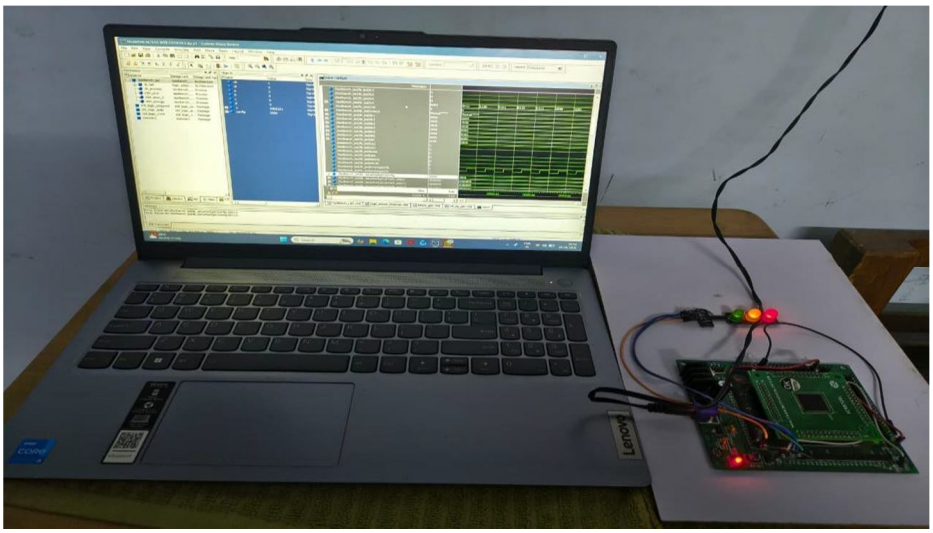


Fig 4.1 FPGA Implementation Output Result

Simulating a clock generator in ModelSim involves creating a clock signal that oscillates between logic levels (0 and 1) at a specified frequency. This is an essential step in digital design, as the clock drives the timing of the entire circuit. To simulate a clock generator in ModelSim, you typically write a testbench in VHDL or Verilog that includes a clock signal definition.
The clock is generated by toggling a signal between high (logic 1) and low (logic 0) at regular intervals, using a process or an always block. For example, in Verilog, you might use a forever loop with a specified time delay to continuously flip the clock signal. Once the clock generator is defined in the testbench, you compile the testbench and run the simulation in ModelSim.

B. *Corruption Attack*

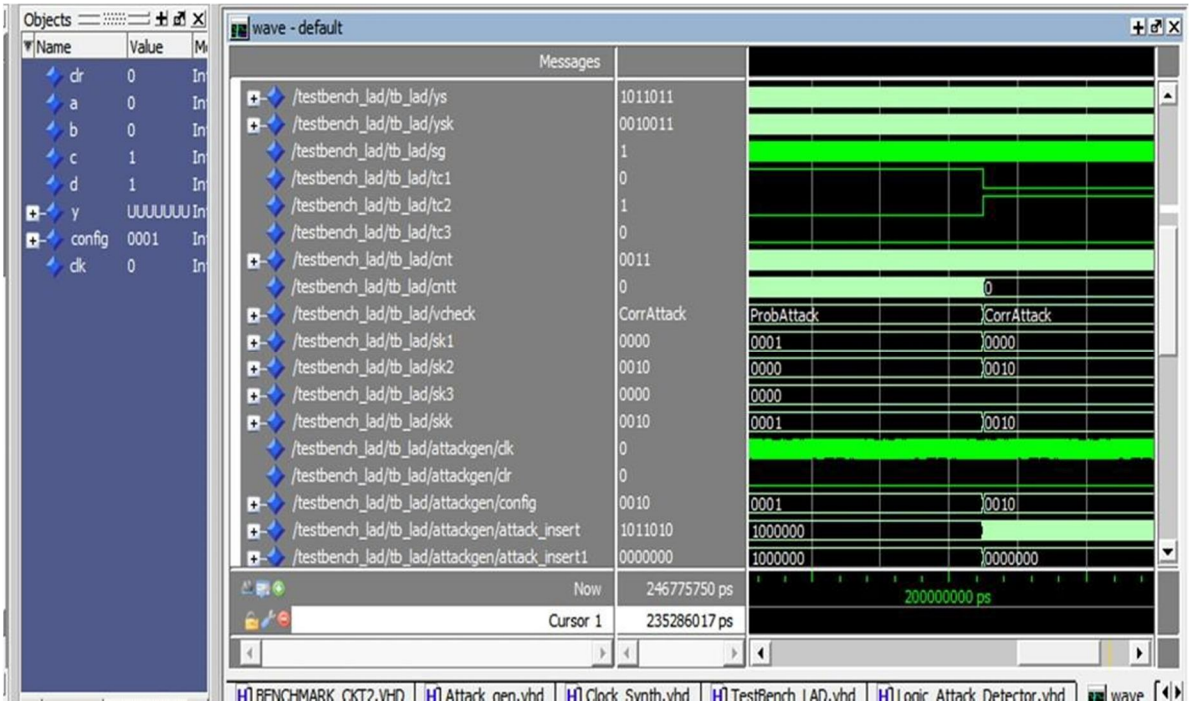


Fig 4.2 Corruption Attack

Detecting a corruption attack in a logic locking system through simulation in ModelSim involves verifying that the circuit's correct functionality has been compromised due to unauthorized tampering or attacks aimed at bypassing the locking mechanism. Logic locking is a hardware security technique used to protect intellectual property by adding extra logic gates or key inputs into the design, requiring a specific secret key to unlock the correct functionality of the circuit. In the case of a corruption attack, the attacker tries to alter the circuit, modify the key, or manipulate the design to gain access without the proper key.

To simulate the detection of such an attack in ModelSim, the locked circuit (along with its testbench) is created, and various key inputs, including incorrect and corrupted ones, are applied.

The testbench would compare the outputs generated by the circuit for both valid and tampered keys to determine if the design exhibits faulty behavior, such as incorrect logic states or unexpected outputs.The testbench would compare the outputs generated by the circuit for both valid and tampered keys to determine if the design exhibits faulty behavior, such as incorrect logic states or unexpected outputs

C. Probing Attack

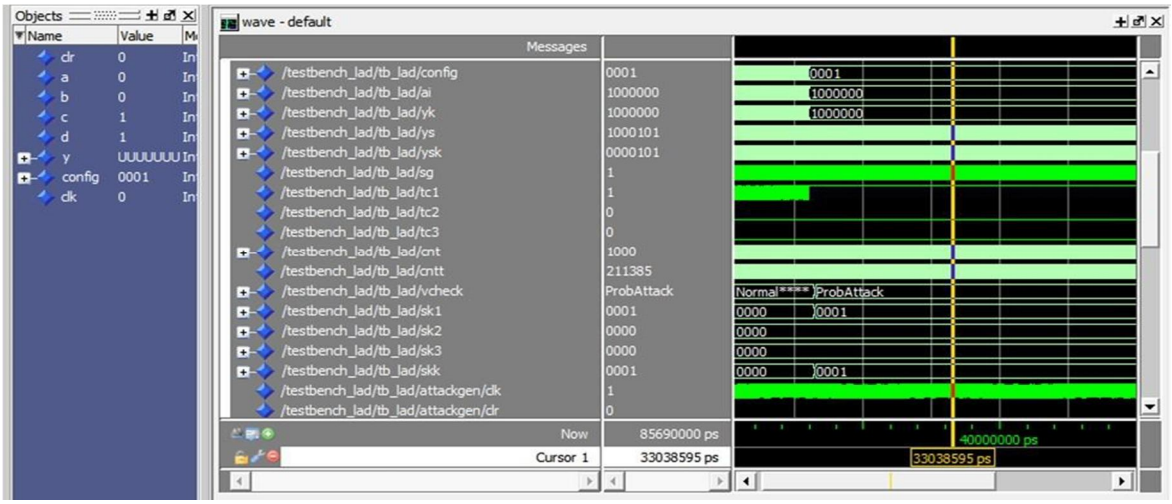


Fig 4.3 Probing Attack

Detecting a probing attack in a logic locking system through simulation in ModelSim involves monitoring the circuit's response to potential attempts by an attacker to extract sensitive internal signals or key bits. A probing attack targets the internal workings of the hardware, trying to directly observe the values of signals at critical points to reverse-engineer the locking mechanism or determine the secret key.

D. Normal Process

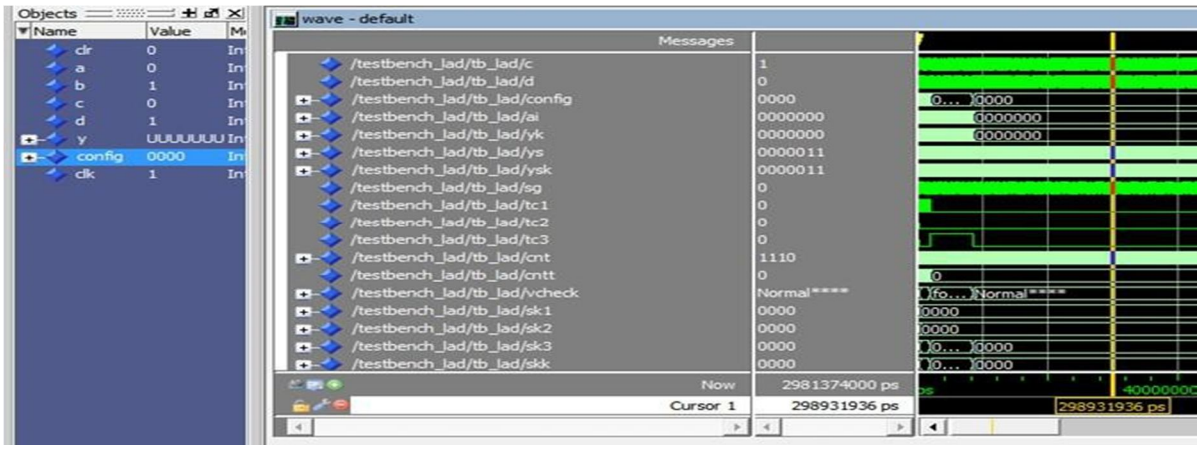


Fig 4.4 Normal Process

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VII July 2026- Available at www.ijraset.com

Dataflow Diagrams

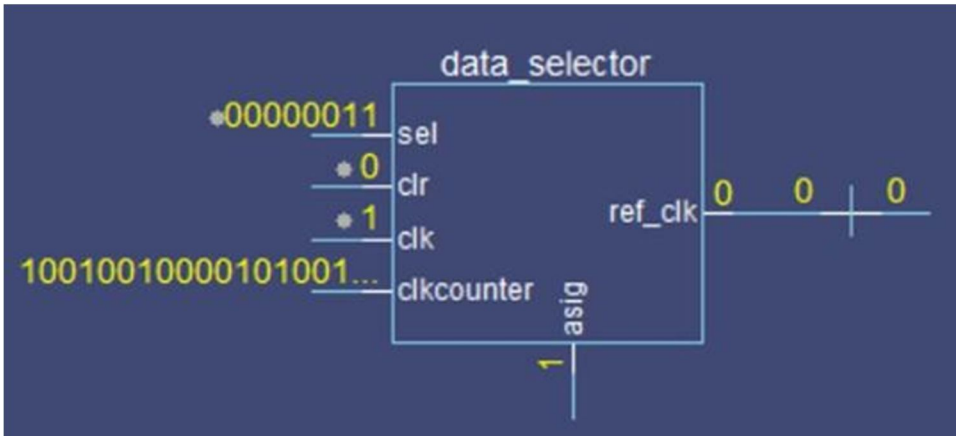


Fig 4.5 Dataflow Diagrams

E. Testing And Validation

Testing procedures for VLSI (Very Large Scale Integration) code involve a series of steps to ensure that the designed integrated circuits (ICs) meet the required specifications and function correctly.

1) Timing Verification

Pre-simulation checks:

Syntax errors: Ensure that the HDL code adheres to the syntax rules of the language being used (e.g., Verilog, VHDL). Common syntax errors include missing semicolons, incorrect module port declarations, and misspelled keywords.

Logical errors: Verify the logical correctness of the design by reviewing the RTL (Register Transfer Level) code. Check for unintended behavior, incorrect data paths, and functional inconsistencies.

Functional simulation:

HDL testbenches: Develop testbench code written in the same HDL as the design to verify its functionality. Testbenches generate stimulus (input vectors) to apply to the design and monitor the responses (output signals).

Operational scenarios: Design test cases to cover various operational scenarios, including normal operation, boundary conditions, error conditions, and corner cases. Stimulate the design with input vectors that exercise different paths and functionalities.

Code coverage analysis:

Code coverage metrics: Utilize code coverage analysis tools to measure the extent to which the design code has been exercised by the testbench. Common code coverage metrics include statement coverage, branch coverage, and condition coverage.

2) Functional Verification:

Gate-Level Simulation:

Functionality Verification: Gate-level simulations verify the correct functionality of the design after synthesis and optimization. The synthesized netlist, which consists of gates and flip-flops, is simulated to ensure that it behaves as intended.

Design for Testability (DFT):

Scan Chains: DFT techniques like scan chains enable efficient testing of integrated circuits by serially shifting in test patterns and capturing output responses. Scan chains facilitate scan-based testing, which is widely used in manufacturing testing.

Built-In Self-Test (BIST) Circuits: BIST circuits are embedded within the design to perform self-testing without requiring external test equipment. BIST circuits generate test patterns, apply them to the circuit under test, and analyze the responses to detect faults.

Fault Simulation:

Stuck-At Faults: Fault simulation involves injecting various fault models into the gate-level netlist to evaluate the design's robustness against manufacturing defects. Stuck-at faults simulate permanent faults where a signal is stuck at a constant logic value (0 or 1).

Bridging Faults: Bridging faults simulate shorts between two or more signal lines, potentially causing unintended connections and logical errors. Fault simulation assesses the design's ability to detect and tolerate bridging faults.

V. CONCLUSION

A. *Conclusion*

Side-channel attacks pose significant threats to System on Chip (SoC) platforms, particularly affecting the operations of FPGAs. The increasing integration of diverse functionalities onto a single silicon platform in modern chip designs has heightened the need for robust security measures, particularly encryption. Any side-channel attacks targeting FPGA devices can compromise the system's integrity by manipulating logic operations, leading to unintended data leaks from external sources. In the proposed system, specialized test circuits are developed to closely monitor and analyze the impact of side-channel attacks.

The system model features a configurable high-speed clock switching network (HSwNw) that operates with multiple clock sources to simulate various attack scenarios. Additionally, a differential attack generator is integrated to introduce side-channel, corruption, and FPGA replay attacks into the application circuit for testing. The system's performance is evaluated continuously through a Recurrent Pattern Verification (RPV) Algorithm, which classifies the effects of each attack.

The evaluation includes metrics such as power consumption, latency, and device utilization. Cryptographic principles and protective frameworks are typically employed to secure most SoC platforms, but under induced attacks, a leakage power of 0.009 watts was observed, demonstrating the system's vulnerability and the importance of countermeasures.

B. *Future Scope*

The proposed hardware Trojan detection framework for FPGA-based systems presents several promising directions for future research and development. Future work may focus on extending the detection mechanism to support a wider range of attack vectors, including fault injection attacks and power analysis attacks, to build a more comprehensive security system. The integration of deep learning and neural network-based anomaly detection algorithms can further enhance detection accuracy and reduce false positive rates. Additionally, the framework can be scaled and optimized for deployment in resource-constrained embedded systems and edge computing environments where security remains a critical concern. Future research may also explore the use of formal verification methods in conjunction with dynamic monitoring to provide stronger security guarantees. Expanding compatibility with heterogeneous SoC architectures and cloud-based FPGA platforms such as AWS and Azure represents another valuable direction.

Furthermore, the development of automated Trojan insertion and testing tools will support more rigorous benchmarking and evaluation of detection systems, ultimately contributing to the establishment of standardized hardware security protocols and methodologies

VI. ACKNOWLEDGEMENT

We are very truly indebted to Late Col. Dr. Jeppiaar, M.A., B.L., Ph.D., Founder, Jeppiaar Educational Trust, Dr. M. Regeena J Murali, B.Tech, M.B.A., Ph.D., our Chairman and Managing Director and our Principal Dr. K. Senthil Kumar, M.Tech., Ph.D., FIE, MISHREA, MISTE for their blessings and the facilities to carry out the project.

We would like to express our sincere gratitude to Dr. J. Jebastine, M.E., Ph.D., Professor and Head of the Department of Electronics and Communication Engineering. We thank our project coordinator Dr. J. Jebastine, M.E., Ph.D., We are thankful to our project supervisor Dr. G. Balachandran, M.Tech.,Ph.D, for giving valuable suggestions and guiding us in all stages of the project. We finally thank all the faculty members of the Department of Electronics and Communication Engineering for their constant support.

We would also like to thank our parents and friends for all their support and we thank the Almighty God for helping us complete this project successfully.

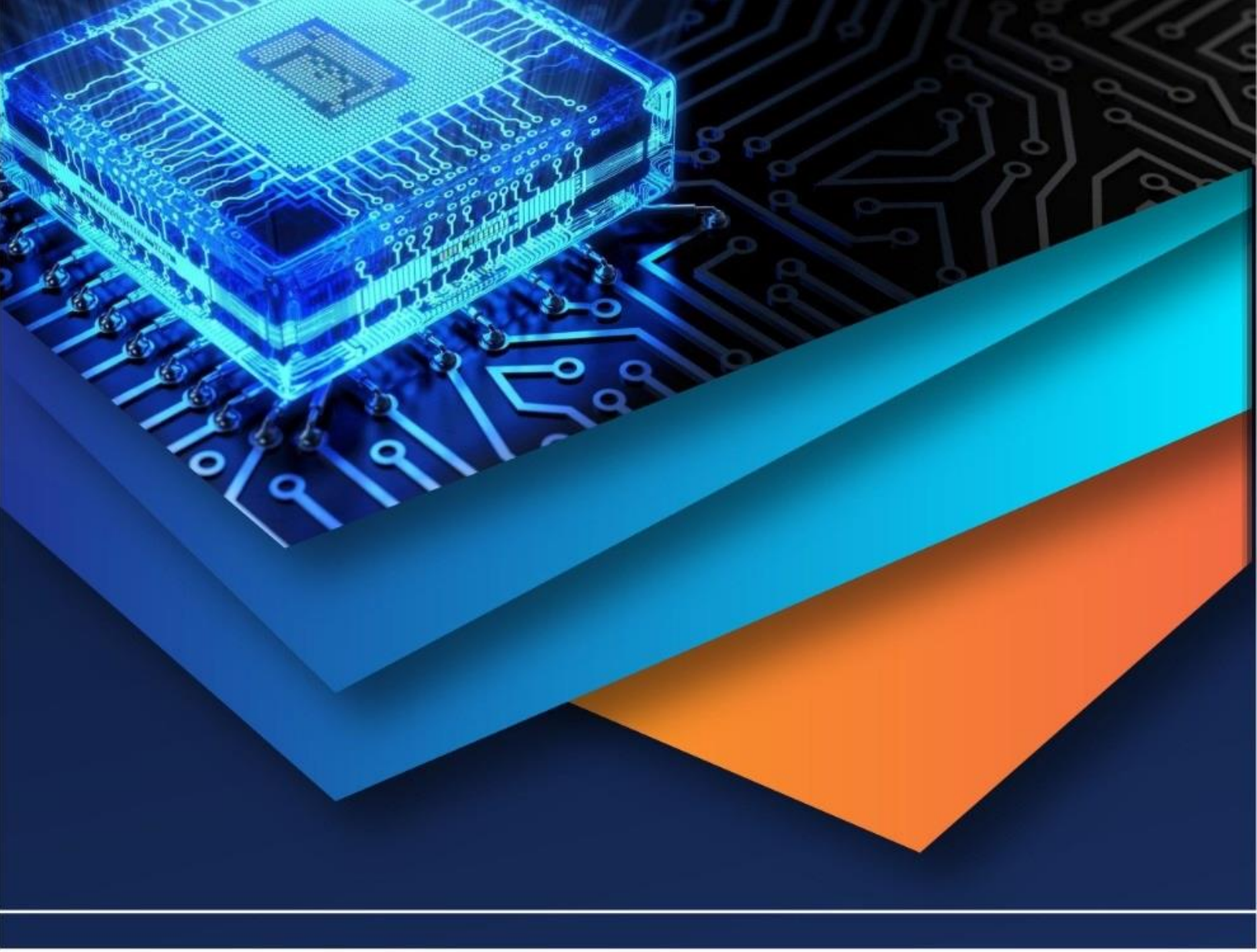
REFERENCES

[1] Rao Bommana, S. Veeramachaneni, S. Ershad, S. and Srinivas, M. B. (2025) "Mitigating Side Channel Attacks on FPGA through Deep Learning and Dynamic Partial Reconfiguration", Scientific Reports, vol. 15, Article 13745.
[2] Zoni et al, D. (2025) "An FPGA-Based Open-Source Hardware-Software", IEEE Transactions on Computers.
[3] Kat-te, S. and Fernandez, K. E. (2025) "SoK: Trusted Execution in SoC-FPGAs," arXiv preprint.
[4] Sergej Meschkov, Daniel Lammers, Mehdi B. Tahoori, Amir Moradi, (2025) "Design and Implementation of a Physically Secure Open-Source FPGA and Toolchain", CHES.
[5] Q. Hou, Z. Liu, Z. Yang and C. Yang, (2024) "Hardware Trojan Attacks on Reconfigurable Interconnections of FPGA-Based CNN Accelerators and a PUF-Based Countermeasure", Micromachines, vol. 15, no. 1, 149.
[6] X. Li, S. Chai, L. Wang and H. Wang, (2023) "A Configurable and Automated Testing Framework for Hardware Trojan Detection in FPGAs", in Proc. WCNA, LN EE.
[7] Jain, A. Zhou, Z. and Guin, U., (2023) TAAL "Tampering Attack on Any Key-based Logic Locked Circuits". ACM journals, 26(4), pp.1-22.
[8] Shamsi, K. Meade, M. Li, T. Zhao, Z. Pan, D. Z and Jin, Y. (2023) "Cyclic obfuscation for creating SAT-unresolvable circuits," in Proc. Great Lakes Symp.

VLSI, pp. 173–178.

[9] Ribes, S. Malatesta, F. Garzo, G. and Palumbo, A. (2022) "Machine Learning-Based Classification of Hardware Trojans in FPGAs Implementing RISC-V Cores".

[10] Ahmed, Q. Wiersema, T. and Platzner, M. (2022) "On the Detection and Circumvention of Bitstream-Level Trojans in FPGAs", IEEE ISVLSI.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY