An Efficient FPGA Based Hardware Trojan Detection Model Using Partial Reconfiguration Procedure

Kalakurasa Rakesh[1], Moturi Satyanarayana[2]
[1]*Associate Professor, MVGR College of Engineering*
[2] *Professor, MVGR College of Engineering*

Abstract: The use of Field Programmable Gate Array (FPGA) became a fundamental device in modern system on chip (SoC) designs. Recent reports about hardware Trojan attacks on secure system are on upswing due to the outsourced designs with third-party IP cores. The globalization of hardware supply from manufacturers reduces NRE cost but increases security challenges in the form of Hardware Trojans which may put the end user confidential date at risk. The necessity to detect Hardware Trojan by traditional methods like Trojan free gold chips always faces scalability limitations by relying solely on chip sequences. The recoverable methods are mostly based on the identification of the Trojan presence at behavioral level. In this paper a new technique based on partial reconfiguration procedure was introduced for the Trojan insertion and to detect it without a golden chip. Experimental data obtained from the PYNQ Z2 board shows that this work based on dynamic partial reconfiguration uses lowest resource utilization and the detection accuracy is noteworthy.

Index Terms: Third-party IP cores, Dynamic Partial Reconfiguration (DPR), Hardware Trojan and Reconfigurable system.

I. INTRODUCTION

The FPGA market now offers a variety of reconfigurable devices and systems thanks to developments in integration technology. There is fierce rivalry among FPGA manufacturers as a result of the devices' ongoing expansion. Shortening the time to market for the product or device and winning over the competitors has led to shorter design time and the requirement for rapid prototyping. The third-party IP core usage in design always boosts the design cycle, leading to reduced time to market. Furthermore, third-party IP cores from various vendors could also potentially carry a Trojan embedded in the design, disrupt the intended operation of the system, and diminish the trust level of integrated circuit being designed. Malicious code in these chips can lead to unintentional leak of private information in turn allowing unauthorized access. In order to sell the product in the market, the end users must have a guarantee that the designs are hardware Trojan-free and exhibited acceptable behavior. Hardware Trojan is referred to an extraneous logic coupled to the original design of the system in the logic design phase or after fabrication. Due to the high cost of facility maintenance and the ongoing need for improvements, fewer fabrication units are operating globally. A lot of manufacturers depend on these facilities owned by third parties for their manufacturing because the majority of design units lack IC fabrication facility. As system density levels rise, maintenance costs rise and time to market decreases and increased competition from manufacturers, the designers rely on third-party fabrication units for their designs to get fabricated. It is a challenging aspect to narrow down the existence of hardware bugs in form of Trojans inserted inside a design, as these remain hidden deep in the logic designed for the system functionality. There are different approaches proposed for revealing the source and type of hardware Trojans. By analyzing the behavior of golden chips, estimating hardware defects by estimating thermal maps, electromagnetic leakage, or any operational changes will determine whether a Trojan is present in a circuit. Most of the methods compare the operational characteristics of the affected chip designed with a golden-chip known as a Trojan free chip for the output sequence [18]. As the complexity of the design requirements increases, these golden chips become increasingly rare and expensive to use for all fault identification.

II. RELATED WORK

The hardware Trojan identification approaches are grouped into two major categories like side channel methods [5] and practices. Architectural methods utilize Trojans by using the underlying architecture to identify

and maximize the probability of finding Trojans during evaluation[19]. Lingxi Liao et al. attempted to find the Trojan movement by utilizing information from electromagnetic side-channel leakage[6]. Gor Piliposyan et al. introduced a method in which a Trojan is detected by differential power monitoring [4], in which the changes in power utilization on gates are selected to detect the Trojans. Side channel methods use a different method in which they localize the effect of a Trojan without triggering them. This approach looks at how Trojan components might affect circuits, considering things like delay, power use, and quiescent current, helping us detect their presence. In his paper [8], Y. Jin demonstrated the gate level characteristics tractable for quite a few time-critical constraints, including path delay fingerprints and power leakage. Path delay fingerprinting is one of the earliest side-channel techniques [8]. The technique consists of measuring the path delay data of integrated circuits and comparing them to a validated "golden" version. Path delay fingerprinting could detect Trojans impacting circuit timing extremely effectively, however it is constrained due to requiring a trusted reference chip. Potkonjak et al. [10] created a gate-level characterization scheme, where devices acquire gate-specific data, such as delay and power leakage, to find Trojans. This gate-level characterization gives precise localization, but as IC designs become more complicated the scalability is an issue. The authors Koushanfar et al. [9], introduced holistic sub modular framework which enhances robustness to environmental and measurement noise to a detection system. The framework includes measurements like delay, power current and unifies them into one modular framework to enhance detection accuracy. The modularity helps the model generalize better across different Trojan models and manage variations in the fabrication process. Furthermore, due to the simultaneous increase in the volume of logic gates and the scaling in size for all of the device parameters, gate- level characterization was restricted to specific dimensions. Third party IPs have lots of availability and can be utilized to design quickly but will reduce the data security associated with the end user. Rajendran et al. [11] proposed another type of side-channel method that leverages frequency. They proposed a ring-oscillator-based design-for-trust (DfT) architecture. The authors introduce ring oscillators in the chip and utilize the ring oscillators in the frequency monitoring to observe frequency changes that occur with the addition of a Trojan. While the proposed technique enhances the Trojan detection rate through the process of on-chip monitoring, it still relies on a trusted baseline and comparison to identify Trojans. Architectural methods allow greater flexibility in Trojan mitigation, especially in FPGAs. Unlike other computing platforms, DPR allows for these modules to be replaced at runtime without interruption of the system. Kim and Villasenor [12] suggested DFR to protect SoCs by mixing the bus signals dynamically. In a follow-up work [13], they introduced a secure SoC bus architecture incorporating an arbitrator, multiplexer, and address decoder. The implementation includes cyclic redundancy check (CRC) circuits and output voting to select the correct data path among numerous IP variants, to improve resilience to Trojan activation. Al-Anwar et al. [14] showed a Trojan detection policy for reconfigurable logic fabrics. This method focuses on identifying problems that are unique to cyborgs. The authors' method combines the concepts of signal monitoring with spatial and temporal correlation, utilizing the partial-reconfiguration capability of the FPGA in order to isolate and disable the infected resources. The architectural change and side channel analysis requires a golden IC based model for verification which have a limited availability in the market. Simply by using a third party core, for example, this will make the chip design more vulnerable to attacks from hardware Trojans because these cores will not have the Trojan detection capability, and thus will not be trusted. FPGA technology is renowned for its quick prototyping and small-unit system implementation. The advancement of FPGA technology has made it possible to use them in a wide variety of applications by supporting partial reconfiguration and dynamic partial reconfiguration (DPR) of a function while it is running. A small portion of the FPGA core can be reconfigured in the DPR facility without affecting the original design. In (12), Lok-Won Kim explained the Dynamic Function Replacement (DFR) for SoC security against hardware-based attacks, which provides direct multiplexing of bus signals. Chakraborty et al. [15] showed that it is possible to insert a hardware Trojan directly into an FPGA by changing its configuration bitstream. This method avoids the usual detection techniques. This approach highlights a critical vulnerability in reconfigurable devices, especially when third-party IP

or unverified bitstream's are involved. The research exposed the underexplored security threats embedded at the configuration layer, underlining the importance of secure bitstream management and validation. To mitigate such threats, dynamic partial reconfiguration (PR) has emerged as a defense mechanism, allowing sections of an FPGA to be reprogrammed during runtime without halting overall system operation. The Xilinx partial reconfiguration design flow [16] provides practical guidelines for secure and efficient reconfiguration in real-world systems. Leveraging PR enhances hardware adaptability, permitting infected regions of an FPGA to be dynamically replaced or isolated. Advanced AMBA bus architectures provide universal communication among different functional modules of the system. In [13], a modified bus architectures featuring an arbitrator, bus mux, and address decoder are described. Multiplexing the outputs of reconfigurable IP blocks, performing CRC on Trojan detection, and using the Multiple Variants method, this allows for the dynamic replacement of multiple IPs output from various blocks. Trojan triggering techniques have also evolved, making detection more difficult. Hsu et al. [17] proposed a Trojan detection method aimed at countering balanced controllability-based triggers. This type of stealth activation allows the attacker to design the Trojan to avoid unusual behavior during standard testing. Their detection approach analyzes signal control patterns to expose unusually balanced paths, which can be indicative of dormant Trojan logic. In a more futuristic approach, John et al. [19] introduced the concept of Quantum Trojan insertion, where activation is governed by quantum conditions, allowing ultra-covert circuit manipulation. This represents a new class of threats, integrating quantum theory with digital hardware design, potentially making traditional side-channel and architectural defenses ineffective. Although this is still a theoretical stage, the paper highlights the need for active defense against new threat models. These contributions show the changing field of hardware Trojan detection. FPGA-based systems, while offering flexibility and reconfigurability, introduce novel threat surfaces such as bitstream manipulation [15] and quantum-controlled triggers [19]. At the same time, the adaptability provided by partial reconfiguration [16], [18] presents promising architectural countermeasures. Emerging detection methods that focus on improved trigger mechanisms [17] show the need for continued progress in both modeling and mitigation strategies.

III. HARDWARE TROJAN CLASSIFICATION

To effectively implement a method for identifying hardware Trojans, it is needed to first establish a clear classification of the different hardware Trojan variants. The classification is based on physical size, activation methods, and action features. The physical internal characteristics are further divided into size, its type, functional distribution and placement of modules inside the structure. The activation characteristics describe the mechanism by which a hardware Trojan is triggered, initiating its malicious behaviour. These characteristics are typically categorised as either internally activated or externally activated, depending on how the Trojan is set into operation. Action features provides information of the type of change applied to the original system design. Figure 1 below shows a complete Trojan classification.

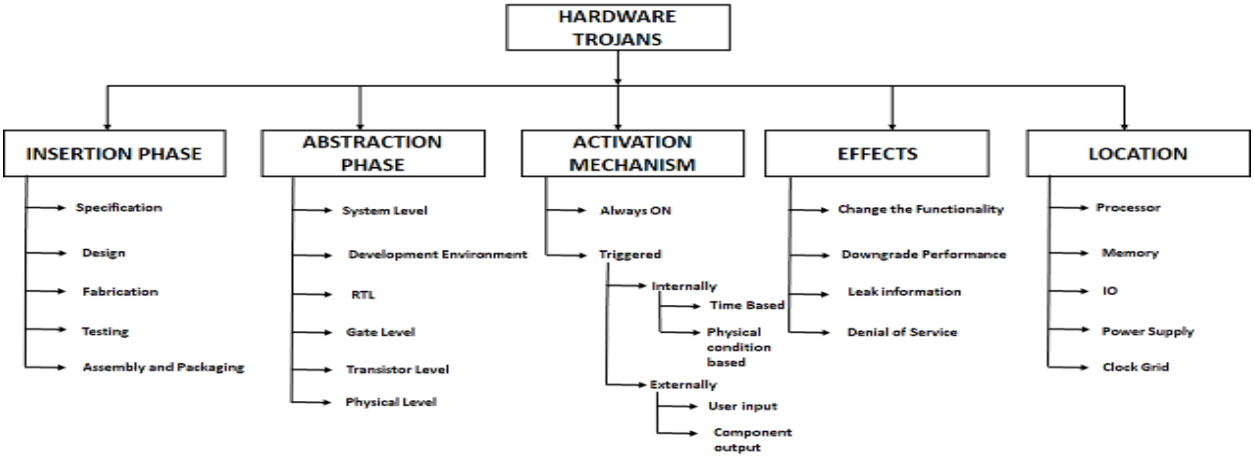

Fig- 1. Hardware Trojan Classification

Several Trojan classifications available can be grouped based on the trigger or activation mechanisms. The classification above assumes that Trojans are added during the fabrication stage. However, the insertion of Trojans is possible at various design stages and at the final stage of manufacturing also. The Trojan insertions fall into the first phase in which the designer can add a hardware-Trojan by changing the original functional logic of the system. During validation, the test samples can miss detecting Trojans without identification. At the Abstraction, system, gate, physical, or developmental levels, Trojans can be inserted. Hardware Trojans are categorized based on the negative impact they have on system functionality. These impacts may vary from subtle functional disruptions to complete system breakdown. When a Trojan is embedded deep within the core components, it can alter essential functions, reduce overall performance, and potentially lead to the exposure of sensitive user information. Whether located in a single spot or spread across various parts of a circuit, the Trojan can influence the system in a similar manner. These malicious components may operate independently or collectively, disrupting system operations at multiple levels. Identifying a Trojan involves not only detecting its presence but also pinpointing its exact location, as this plays a key role in understanding and mitigating its effect on the system. Consequently, the position of the Trojan within the hardware is a crucial factor in its classification.

IV. TROJAN INSERTION

Hardware Trojans usually have mainly two parts: triggers & payloads. The bit sets of trigger define the exact Requirement for Trojan activation, like a specific combination of input bit sequences or signals. Once activated, the payload introduces malicious behavior, which includes actions such as causing a system crash (resulting in service unavailability or denial of service) sensitive date leaking, such as cryptographic keys, through hidden channels like wired serial communication. A hardware functional manipulation in the form of a malicious Trojan can be introduced at various levels of system design or during manufacturing.

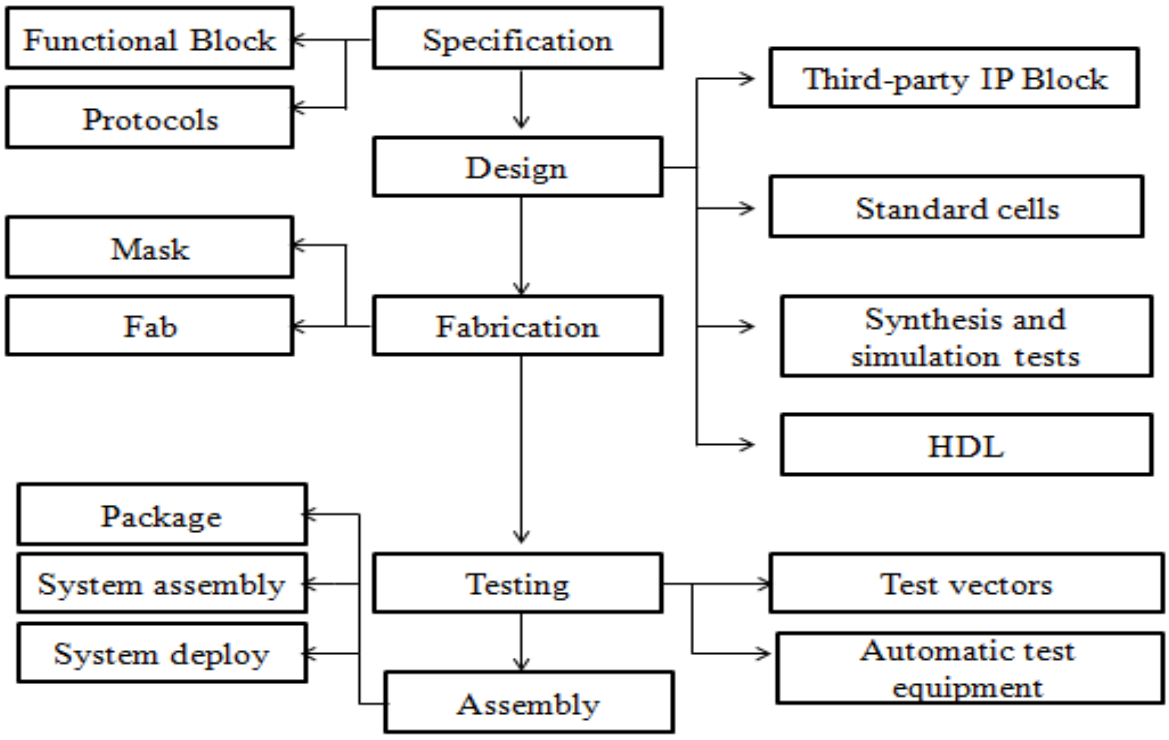

Fig- 2. IC development lifecycle

In the fig 2 shows the development life cycle of a system and its vulnerable stages where a Trojan can be inserted.

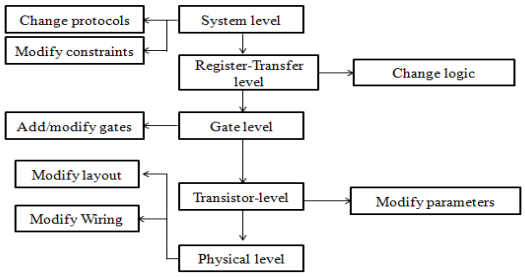

Fig 3. Different Levels of hardware Design.

Fig 3 shows various levels of IC design where the Trojan circuits can be incorporated. The middle boxes show top down VLSI flow, while the left and right boxes display Trojan at that particular level. In this research work, we introduced three different Trojan's based on their trigger-input. These Trojans are initiated on different intervals have the influence on the original circuit functionality. The three types of Trojans are i) Always ON type, ii) Condition Based Type and iii) User Input Type. Each type has the same goal: to change the original functionality or to leak data. The designed Trojan types have effects on the original design logic which include i) Change in underlying logic Function and ii) Denial-of-Service. The change in logic output affects the original behavior of the intended logic.

Trojans can disrupt the entire system by limiting or halting its operations, ultimately leading to a failure in achieving the expected performance. This disruption is commonly referred to as a Denial-of-Service (DoS) attack. During the design stage, attackers may alter the system's internal logic by embedding harmful code that avoids detection during routine testing. In complex system architectures, it becomes difficult to create and verify test cases for every possible scenario. This loophole allows attackers to design Trojans that activate only under rare input conditions, helping them stay hidden and inactive during typical testing processes.

V. DETECTION METHODOLOGIES

The detection methods for the Trojan require the localization of hardware Trojan which is a complex issue and studying their influence on the overall design are main focused here. In a System-on-Chip (SoC) containing multiple functional logic modules, an efficient interconnection mechanism that facilitates data transfer and communication between the modules is critical. A system bus serves the purpose of connecting different resources in a system and plays a crucial role in SoC design. The allocation of memory and other shared resources for a functional unit will be handled by the system internal bus.

In general, a bus-master manages the bus when multiple requests come in to access a certain block. This request occurs after decrypting the master module give the signals to reach the Subordinate module, creating a connection between the modules.

In our design, we investigated the detection of digital hardware-Trojans inserted in the design phase of the system hardware during the development phase. The Trojan is inserted into the bus logic design freezes the system bus upon activation. The Trojan will be activated during a specific event. It will disrupt normal operations by blocking regular bus services. The structure of the proposed model was implemented on the PYNQ Z2 programmable logic block of the FPGA. The schematic editor tool available in the Vivado design suite was used to capture the design, and the proposed design consisted of a counter module followed by adder and subtractor modules designed for a dynamic hardware reconfigurable model, as shown in Figure 4.

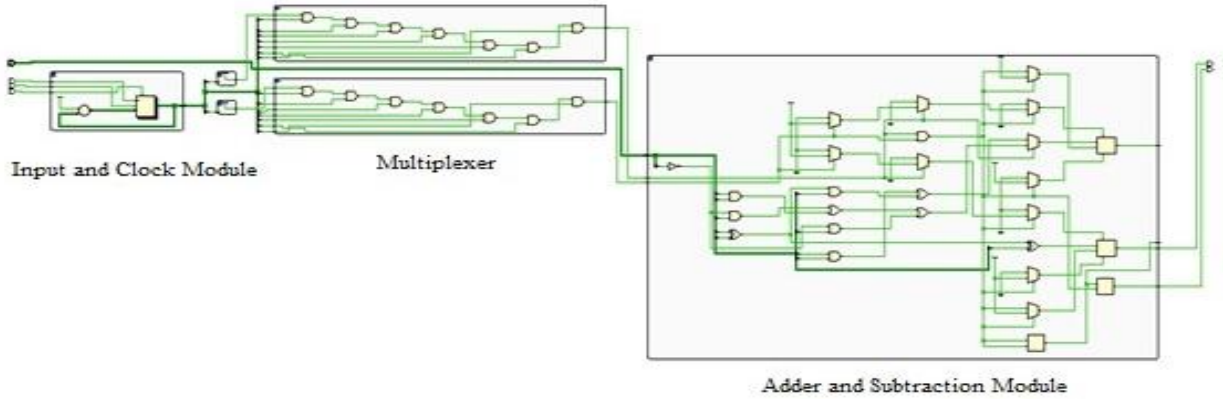

Fig 4. Proposed Dynamic hardware module

The MUX selection lines control the switching between the adder and subtraction modules. The outcomes of the proposed model can be verified by Trojan attack from the simulation output obtained as shown in below figure 5.

The findings indicate that the Trojan becomes active for a short duration when the system bus is temporarily idle, enabling unauthorized control. This limited time window can be exploited to extract confidential information or degrade the system's performance. Despite the brief activation period, the Trojan can cause considerable disruption to the overall process of the circuit.

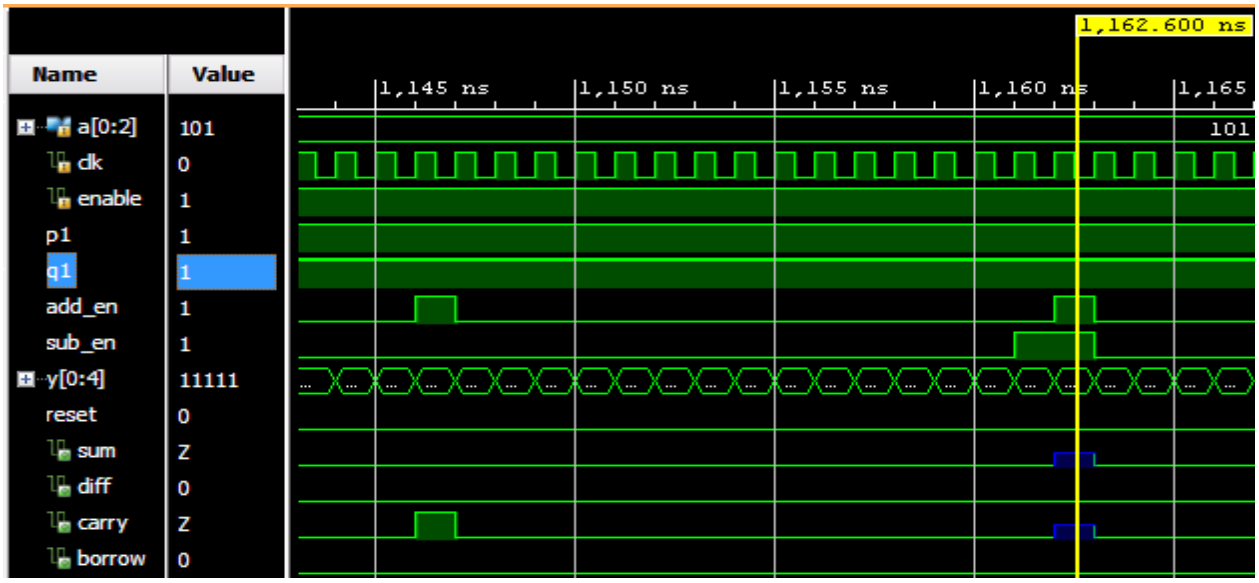

Fig-5. Simulation result for denial of service Trojan

The Trigger-based Trojan has functional changes as evident from below result as soon as it get activated while the system is in use. The simulation results below Fig 6 show its impact.

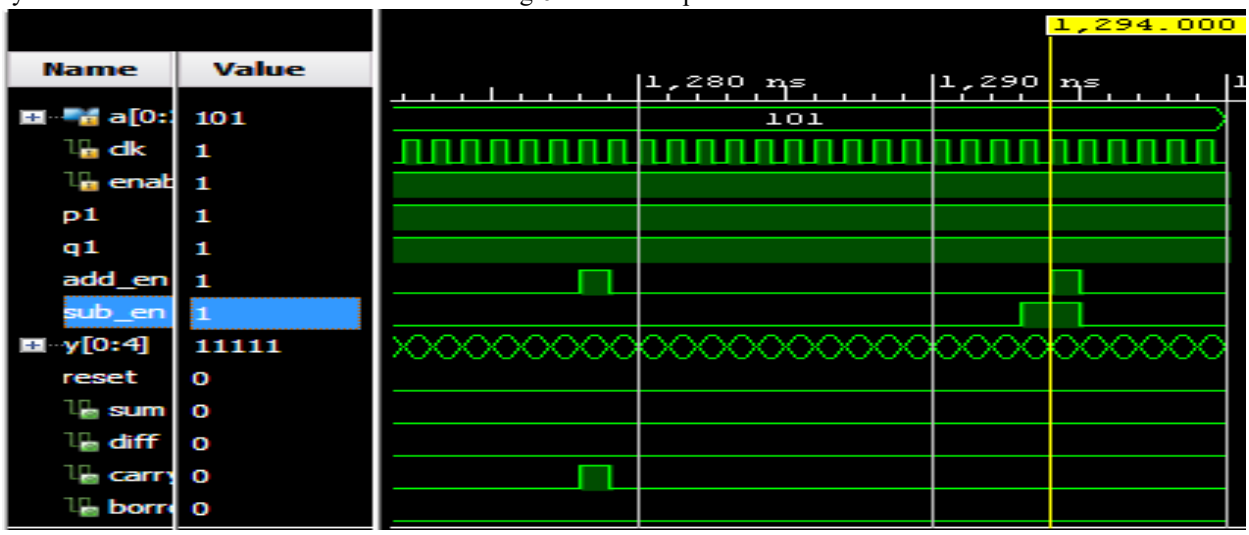

Fig-6. Change in Functionality

The results obtained indicate that the add-en and sub-en signals, when activated to a high state, divert the original function, which is not intended. Detecting a hardware Trojan in a more complex system architecture is difficult. Designers intentionally hide

the bugs so the Trojan can pass through the testing phase without being found.

VI. RECOVERABLE METHODOLOGY

The recovery of the original system functionality after a Trojan attack is vital. System downtime can have a negative impact performance. System recovery can be achieved by leveraging the DPR capability of the Xilinx PYNQ Z2 FPGA. Using DPR, we can modify the Trojan-inserted design dynamically at runtime without altering the functionality of other running modules in the system. Partial reconfiguration in Xilinx FPGA lets us divide the Programmable Logic system into two separate parts. One part can be organized in different ways. A sample design based on Xilinx partial reconfiguration implemented in FPGA is shown below in figure 7.

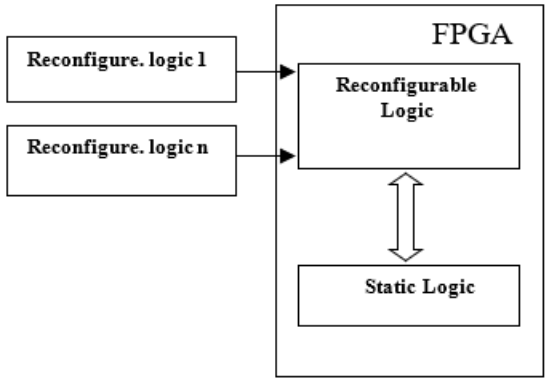

Fig- 7. Xilinx FPGA Partial Reconfiguration Logic Design

With the partial reconfiguration feature of Xilinx PYNQ Z2, we can swap the error-prone module for one that is free of Trojans. This will restore the original functionality to normal without any downtime of the system.

With the help of Xilinx PYNQ Z2 FPGA's PR facility, the error module will be replaced with a Trojan less module, and the original functionality can be restored to normal without any downtime of the system. In our design, we focus on the Trojan insertion and the change in the Trojan module by the partial reconfiguration procedure is successfully verified practically. The overall steps involved during partial reconfiguration are presented in the flow chart in Fig-8.

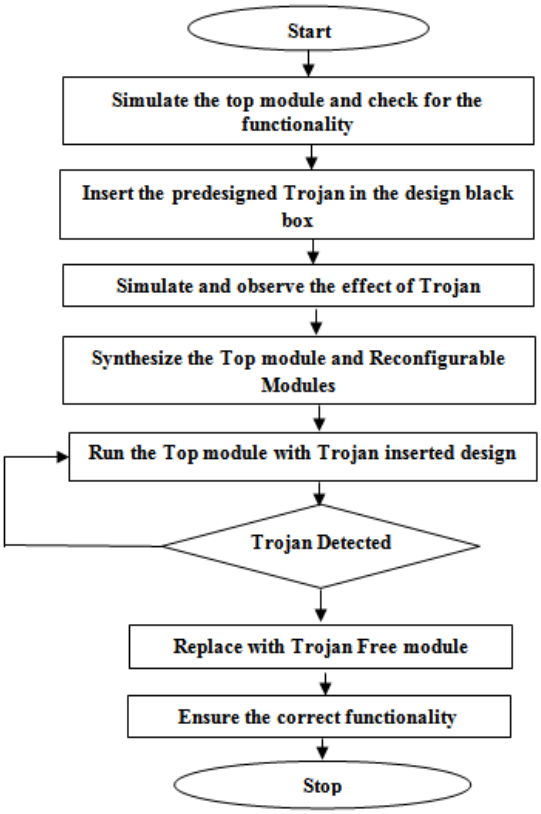

Fig-8. Steps in Partial Reconfiguration

The influence of Hardware Trojans on a system may affect multiple system constraints. Among them, we can see the overall power utilization factor. It shows the changes in the total power used by the top module. The logic power factor is assessed with and without hardware Trojan. The increase in power utilization factor indicates that the system is influenced by Trojan as illustrated in below table 1.

Power Analysis Summary			
Module Tested	Total -Power (Watts)	Junction Temp (°C)	Thermal-Margin (°C)
Main Module	0.748	28.8	47.2
With Trojan	1.214	31.2	62.8
Without Trojan	1.01	28.7	49.9

Table- 1 Power summary of different modules

VII. ANALYSIS

The results obtained from the synthesis show the compatibility of all three modules. In figure 9a, the top and error modules are compatible with each other, and the correct module is also compatible with the original module, as shown in fig 9b. This shows that the replacement of the error module can be performed with the correct module, as both are compatible. The compatibility of both modules is verified using the outputs.

```
DCP1: config_blank/Always_on_top_route_design.dcp
  Number of reconfigurable modules compared = 1
  Number of partition pins compared         = 10
  Number of static tiles compared           = 20
  Number of static sites compared           = 22
  Number of static cells compared           = 32
  Number of static routed nodes compared    = 376
  Number of static routed pips compared     = 353

DCP2: config_error/error_route_design.dcp
  Number of reconfigurable modules compared = 1
  Number of partition pins compared         = 10
  Number of static tiles compared           = 20
  Number of static sites compared           = 22
  Number of static cells compared           = 32
  Number of static routed nodes compared    = 376
  Number of static routed pips compared     = 353
INFO: [Vivado 12-3253] PR_VERIFY: check points config_blank/Always_on_top_route_design.dcp and config_error/error_route_design.dcp are compatible
```

Fig.9.a Synthesis results showing compatibility of Top and error Modules

The system compatibility shows that the proposed model and the Trojan model are interchangeable.

```
DCP1: config_blank/Always_on_top_route_design.dcp
  Number of reconfigurable modules compared = 1
  Number of partition pins compared         = 10
  Number of static tiles compared           = 20
  Number of static sites compared           = 22
  Number of static cells compared           = 32
  Number of static routed nodes compared    = 376
  Number of static routed pips compared     = 353

DCP2: config_correct/correct_route_design.dcp
  Number of reconfigurable modules compared = 1
  Number of partition pins compared         = 10
  Number of static tiles compared           = 20
  Number of static sites compared           = 22
  Number of static cells compared           = 32
  Number of static routed nodes compared    = 376
  Number of static routed pips compared     = 353
INFO: [Vivado 12-3253] PR_VERIFY: check points config_blank/Always_on_top_route_design.dcp and config_correct/correct_route_design.dcp are compatible
```

Fig 9.3(b) Synthesis results showing compatibility of Top and correct modules

From the power utilisation metric, the total on-chip power with the hardware Trojan is estimated as high. Minimum changes can be seen in the power summary as the Trojan designer tries to keep the Trojan hidden. Changing of the Trojan modules with non-Trojan modules has a positive impact on the functional circuit, and hence partial reconfiguration helps in eliminating the hardware Trojans.

VIII. CONCLUSIONS

Hardware Trojans pose a serious threat for the systems that are built mainly on third-party IP cores, downgrading their performances. In this work we presented various stages of hardware-Trojan insertion, its detection, and the recovery of the system using the DPR of the PYNQ Z2 FPGA board. By using the PR of FPGA procedure for replacing the infected logic, we were able to reduce the overall Trojan influence on the original system logic. This technique can be deployed directly in the field, making it highly practical compared to previously discussed solutions. It is also cost-effective and requires minimal expertise to modify the reconfigurable modules.

REFERENCES

[1] Lei Zhang, Youheng Dong, Jianxin Wang, Chaoen Xiao, and Ding Ding, "A hardware Trojan detection method based on the electromagnetic leakage," China Communications, vol. 16, no. 12, Dec. 2019.

[2] Y. Tang, S. Li, L. Fang, X. Hu, and J. Chen, "Golden- chip-free hardware Trojan detection through quiescent thermal maps," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 12, pp. 2767–2779, Dec. 2019.

[3] R. Yasaei, L. Chen, S.-Y. Yu, and M. A. Al Faruque, "Hardware Trojan detection using graph neural networks," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 44, no. 1, Jan.

2025.

[4] G. Piliposyan, S. Khursheed, and D. Rossi, "Hardware Trojan detection on a PCB through differential power monitoring," *IEEE Trans. Emerg. Topics Comput.*, vol. 10, no. 2, pp. 682–693, Apr.–Jun. 2022.

[5] L. Liao and F. Meng, "Hardware Trojan detection and identification using electromagnetic side-channel leakage," in *Proc. 9th Int. Conf. Intell. Comput. Signal Process. (ICSP)*, 2024.

[6] E. Jedari and R. Rashidzadeh, "A hardware Trojan detection method for IoT sensors using side-channel activity magnifier," *IEEE Internet Things J.*, vol. 9, no. 6, pp. 4594– 4603, Mar. 2022.

[7] D. Du, S. Narasimhan, R. Chakraborty, and S. Bhunia, "Self-referencing: A scalable side-channel approach for hardware Trojan detection," in *Proc. 12th Int. Conf. Cryptographic Hardware Embedded Syst. (CHES)*, 2010, pp. 173–187.

[8] Y. Jin and Y. Makris, "Hardware Trojan detection using path delay fingerprint," in *Proc. IEEE Int. Workshop Hardw.- Oriented Secur. Trust (HOST)*, 2008, pp. 51–57.

[9] F. Koushanfar, A. Mirhoseini, and Y. Alkabani, "A unified submodular framework for multimodal IC Trojan detection," in *Proc. 12th Int. Conf. Inf. Hiding (IH)*, 2010, pp. 17–32.

[10] M. Potkonjak, A. Nahapetian, M. Nelson, and T. Massey, "Hardware Trojan horse detection using gate-level characterization," in *Proc. 46th Annu. Design Autom. Conf. (DAC)*, Jul. 2009, pp. 688–693.

[11] J. Rajendran, V. Jyothi, O. Sinanoglu, and R. Karri, "Design and analysis of ring oscillator based design-for-trust technique," in *Proc. IEEE 29th VLSI Test Symp. (VTS)*, May 2011, pp. 105–110.

[12] L.-W. Kim and J. D. Villasenor, "Dynamic function replacement for system-on-chip security in the presence of hardware-based attacks," *IEEE Trans. Reliab.*, vol. 63, no. 2,pp. 661–674, Jun. 2014.

[13] L.-W. Kim and J. D. Villasenor, "A system-on-chip bus architecture for thwarting integrated circuit Trojan horses," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 11, pp. 1921–1926, Oct. 2011.

[14] A. Al-Anwar, Y. Alkabani, M. W. El-Kharashi, and H. Bedour, "Hardware Trojan detection methodology for FPGA," in *Proc. IEEE Pacific Rim Conf. Commun., Comput. Signal Process. (PACRIM)*, Oct. 2013, pp. 177–182.

[15] R. S. Chakraborty, I. Saha, A. Palachaudhuri, et al., "Hardware Trojan insertion by direct modification of FPGA configuration bitstream," *IEEE Des. Test*, vol. 30, no. 1, pp. 45–54, Feb. 2013.

[16] https://www.xilinx.com/support/documentation/sw_man uals/xilinx2017_1/ug947-vivado-partial-reconfiguration- tutorial.pdf

[17] W.-T. Hsu, P.-Y. Lo, C.-W. Chen, C.-W. Tien, and S.-Y. Kuo, "Hardware Trojan detection method against balanced controllability trigger design," *IEEE Embedded Syst. Lett.*, vol. 16, 2024.

[18] S. Murali and K. Rakesh, "Dynamic hardware reconfigurable system using partial reconfiguration procedure of FPGA," *IOSR J. Electron. Commun. Eng.*, vol. 13, no. 3, pp. 01–07, 2018.

[19] J. John, L. Golla, and Q. Wang, "Quantum Trojan insertion: Controlled activation for covert circuit manipulation," *arXiv preprint arXiv:2502.08880*, 2025.

[20] Amr Al-Anwar; Yousra Alkabani; M. Watheq El-Kharashi, "Hassan Bedour Hardware Trojan detection methodology for FPGA", IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM), 2013

AUTHORS PROFILE

K.Rakesh received the bachelor's and master degree in electronics and communication engineering. He is currently pursuing the Ph.D. degree at JNTU, Kakinada. His current research involves the design of reconfigurable SoC Architectures For Drones, Machine Learning for Secure control Applications using IoT.

Dr.**M.Satyanarayana** received his bachelor's degree, master degree and Ph.D degrees in electronics and communication engineering. He is currently a full Professor at MVGR College of Engineering (A), vizianagaram. His research interests include VLSI and Design of Antennas. He has authored or coauthored over 150 publications in peer reviewed journals and conferences.