Hardware Trojans in FPGA Systems: An In-Depth Analysis of Attack Vectors and Security Countermeasures

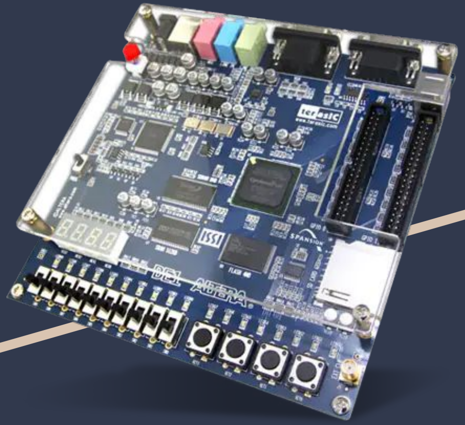
Professor Kose
ECE 213 Hardware Security
David Cai

Outline:

1. Introduction to Hardware Trojans and FPGA Security
2. Types of Hardware Trojans in FPGA Systems
3. Detection Techniques for Hardware Trojans
4. General Security Countermeasures and Best Practices
5. Future Research Directions

Introduction to Hardware Trojans and FPGA Security

So, what are FPGAs?



Field Programmable Gate Arrays

Off-the-shelf

Ability to reconfigure the hardware

Upgrade and bug fixes

Configurable logic blocks (CLBs) and interconnects

Significance

Versatile

Test any number of variables

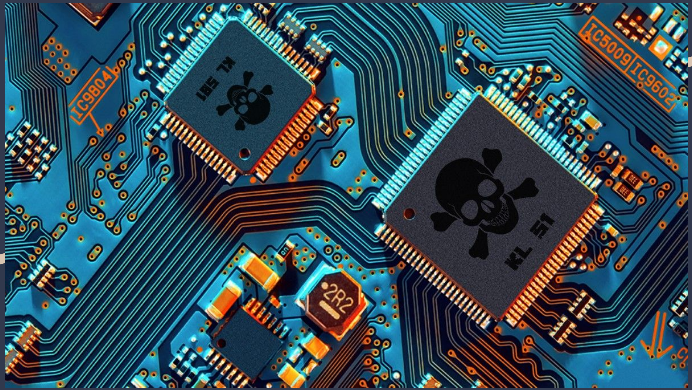
New files transferred onto the device

Incrementally mature the design

Commercial applications

Lower price points

Hardware Trojans?



malicious and hard-to-detect

modification to an electronic circuit or design that causes an electronic devices to behave incorrectly.

Two major components:

trigger logic

payload

Can be introduced in many ways:

1. Design phase: IP cores, backdoors
2. Manufacturing phase: foundry-level attack: rogue employee
3. Supply chain: intercepted shipments or tampered components

Risks and implications of hardware Trojans in FPGA systems

They can:

- Bypass or disable security
- Leak confidential information
- Disable or destroy devices
- Denial of Service (DoS)
- Cost of Detection
- Operational risk and reduced reliability/performance (malfunction)
- Data manipulation

Anyone can access FPGAs

Does not add extra circuitry [parametric] (could add extra circuitry)

FPGAs are made by different people from places

Motivation for studying hardware Trojan attacks and countermeasures

Protect FPGAs

Implement better countermeasures

Improvement upon existing measures

Trusted environment

Types of Hardware Trojans in FPGA Systems

- Functional Trojans
- LUT-Oriented Trojans
- Parametric Trojans

● Functional Trojans

Manipulates operational functionality

Unintended behavior

Active

Triggered under specific conditions

Symptom: data corruption, unauthorized control, or tampered outputs

Primary aspects involve:
Trigger Mechanism: activation: data pattern, timing events, or external signals.

Payload: outcome or effect, from minor functionality changes to severe impacts like complete control over system functions.

A. Murtaza et al., "FPGA Based Intelligent Hardware Trojan Design and its SoC Implementation," *2023 IEEE ICECS*, Istanbul, Turkiye, 2023, pp. 1-4, doi: 10.1109/ICECS58634.2023.10382763.

Functional Trojans “Methodology”

Goal: HT detects and classifies encrypted data streams in an SoC, identifying the source IP core. (sniff the bus)

Required hardware:
SoC is needed (bridge): a RISC-V processor (SweRV EH1) as the CPU with Wishbone interconnect bus was used to develop the full SoC in a shared-bus architecture.

- SoC Setup: Includes UART, SPI, I²C, and AES IP cores connected via Wishbone bus to send signals.

Everything put on: **Artix-7 FPGA (Nexys A7 board)**

A. Murtaza et al., "FPGA Based Intelligent Hardware Trojan Design and its SoC Implementation," *2023 IEEE ICECS*, Istanbul, Turkiye, 2023, pp. 1-4, doi: 10.1109/ICECS58634.2023.10382763.

A. Murtaza et al., "FPGA Based Intelligent Hardware Trojan Design and its SoC Implementation," *2023 IEEE ICECS*, Istanbul, Turkiye, 2023, pp. 1-4, doi: 10.1109/ICECS58634.2023.10382763.

Functional Trojans “Methodology” Cont.

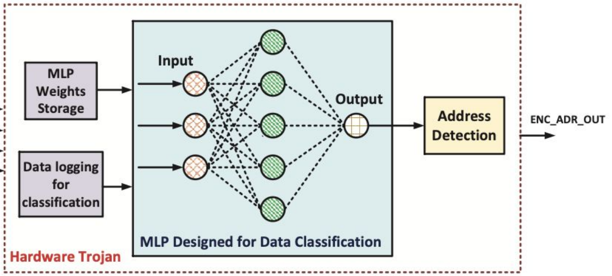


Fig. 2. Proposed Hardware Trojan Design.

- HT Modules:

  - Data Logging: Stores 13 packets for classification.

  - MLP Classifier: Uses pre-trained weights to classify data as encrypted or plain.

  - Address Detection: Outputs AES IP address if data is encrypted.

- Training: MLP trained with TensorFlow; weights stored in memory for real-time use.

- Operation: Logs data, classifies, then outputs AES address if encryption is detected.

A. Murtaza et al., "FPGA Based Intelligent Hardware Trojan Design and its SoC Implementation," *2023 IEEE ICECS*, Istanbul, Turkiye, 2023, pp. 1-4, doi: 10.1109/ICECS58634.2023.10382763.

Functional Trojans “Methodology” Cont.

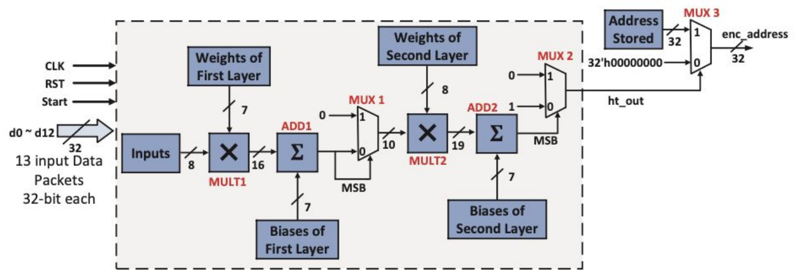


Fig. 3. RTL Design of Hardware Trojan MLP Classifier.

- First Layer: Multiplies inputs by Weights of First Layer, adds Biases, then applies ReLU activation using MUX1.

- Second Layer: Repeats with Weights of Second Layer and Biases, followed by another ReLU via MUX2.

- Classification: MUX3 outputs enc_address if classified as encrypted (ht_out = 1); otherwise, outputs zero.

- Purpose: Detects encrypted data and identifies source IP address in real-time.

Functional Trojans "Results"

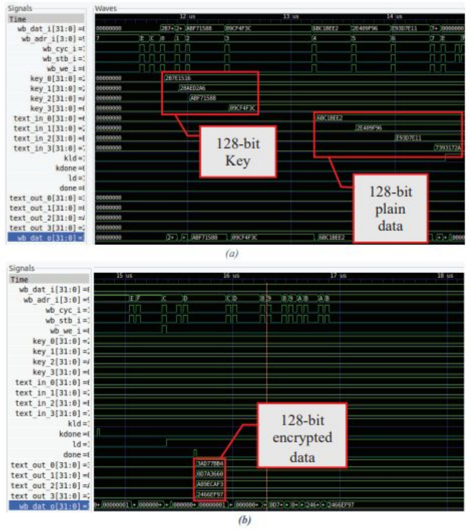


Fig. 4. Simulation Results of AES Encryption.

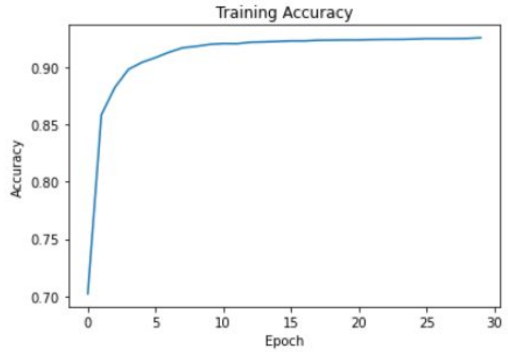


Fig. 5. Epoch vs. Training Accuracy of the Designed MLP.

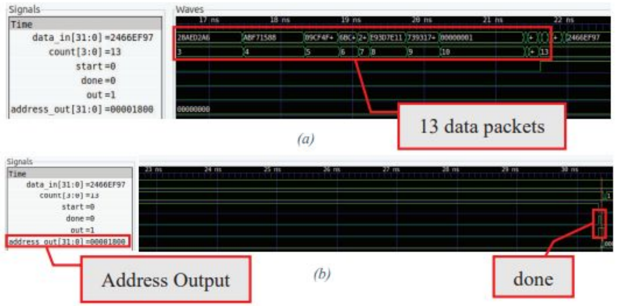


Fig. 6. Functional Verification of Proposed Hardware Trojan IP Core.

A. Murtaza et al., "FPGA Based Intelligent Hardware Trojan Design and its SoC Implementation," *2023 IEEE ICECS*, Istanbul, Turkiye, 2023, pp. 1-4, doi: 10.1109/ICECS58634.2023.10382763.

LUT-Oriented Trojans

What is it?

LookUp Table:

Small memory (stores predefined values or truth tables for input combinations)

In FPGA, fundamental

Logic functions by referencing stored outputs

LUT HT:

In FPGA-based AI modules.

Exploits the structure to alter AI inferences by manipulating outputs in truth tables

No required circuits for triggers and payloads

Hard to detect

Results: falsifying classifications without affecting the circuit area or performance.

Very concerning in AI applications

- LUT Network

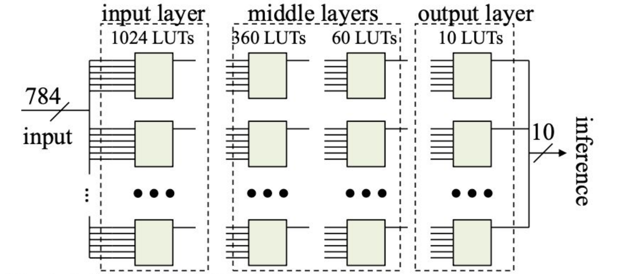


Fig. 1 Outline of LUT-Network.

- **LUT-Network Overview**:
  - AI inference device using **lookup tables (LUTs)**
  - Represents neural network (NN) through **truth tables**
- **Structure**:
  - **Input Layer**: 1024 LUTs for **784-bit input**
  - **Middle Layers**: 360 LUTs + 60 LUTs
  - **Output Layer**: 10 LUTs for **10-bit output**
- **Output Example**:
  - **One-hot Encoding**: Each bit represents a digit
  - **‘0000000100’** = recognized as digit **‘2’**
- **Advantage**:
  - **Low-latency** inference, ideal for **edge AI applications**

LUT Methods

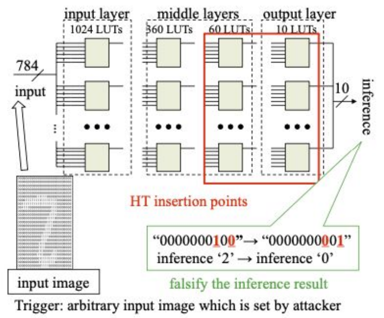


Fig. 2 Outline of the proposed HT.

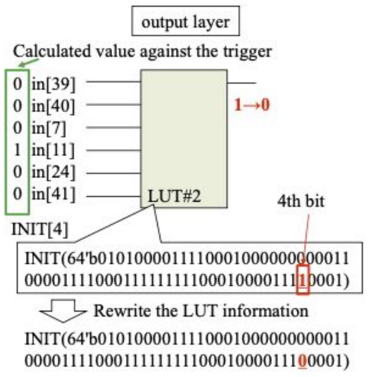


Fig. 3 Example of the proposed HT insertion into LUT#2 in the output layer.

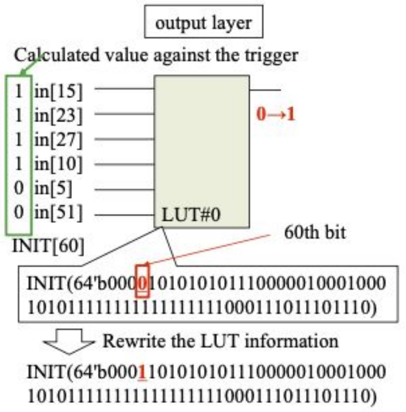


Fig. 4 Example of the proposed HT insertion into LUT#0 in the output layer.

Proposed Hardware Trojan

- **Objective**: Introduce a hardware Trojan (HT) in LUT-based NN on FPGA, targeting inference accuracy stealthily.
- **Trigger Mechanism**: Activated by a specific, attacker-provided input image to alter inference output.
- **Methodology**:
  - Rewrites bits in LUTs, e.g., modifies output from "0000000100" (2) to "0000000001" (0).
  - No additional circuits; relies on modifying LUT initialization (INIT) data.
- **Insertion Points**:
  - Primarily in middle layers to reduce impact on accuracy and evade detection.
- **Stealth**: No added logic or area overhead, making it difficult for traditional detection methods to spot.
- **Complexity:** in middle layers to alter final results.

LUT Results

- **Setup**: Conducted on SASEBO-GII FPGA board with a Virtex-5 FPGA, using MNIST digit recognition.
- **Results**:
  - HT causes intended misclassification from '2' to '0' with trigger image.
  - **Accuracy Impact**: HT inserted into the middle layer: The NN's recognition accuracy dropped only slightly from 92.35% (without HT) to 92.23%.
  - HT inserted into the output layer: The accuracy dropped further to 91.76%.
- **Conclusion**: HT operates effectively with minor accuracy drop or resource usage increase, enhancing stealth and undetectability.

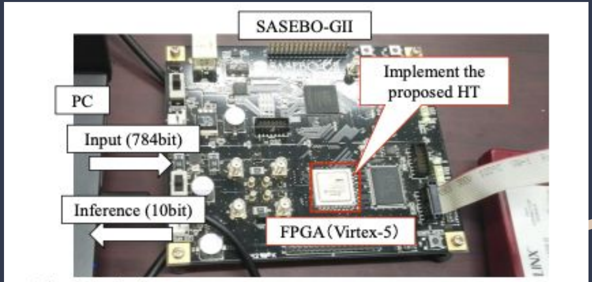
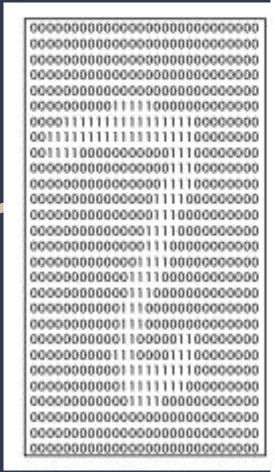
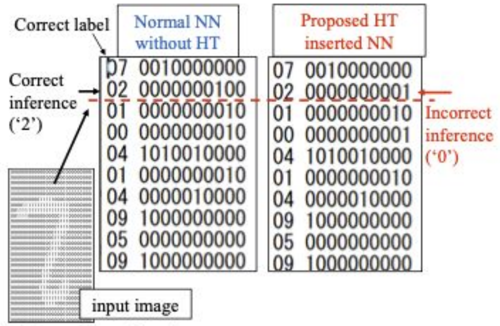


Fig. 6 Evaluation system.


```
0000000000000000000000000000
0000000000000000000000000000
0000000000000000000000000000
0000000000000000000000000000
0000000000000000000000000000
0000000000111110000000000000
0000111111111111111100000000
0011111111111111111100000000
0011110000000000011100000000
0000000000000000011100000000
0000000000000000111100000000
0000000000000001111000000000
0000000000000001110000000000
0000000000000011110000000000
0000000000000011100000000000
0000000000000111100000000000
0000000000001111000000000000
0000000000001110000000000000
0000000000011100000000000000
0000000000011100000000000000
0000000000011000001100000000
0000000000111000011100000000
0000000000011111111100000000
0000000000011111111000000000
0000000000001111000000000000
0000000000000000000000000000
0000000000000000000000000000
0000000000000000000000000000
```



Normal NN without HT		Proposed HT inserted NN	
07	0010000000	07	0010000000
02	0000000100	02	0000000001
01	0000000010	01	0000000010
00	0000000010	00	0000000001
04	1010010000	04	1010010000
01	0000000010	01	0000000010
04	0000010000	04	0000010000
09	1000000000	09	1000000000
05	0000000000	05	0000000000
09	1000000000	09	1000000000

Fig. 8 Experimental results.

Parametric Trojans

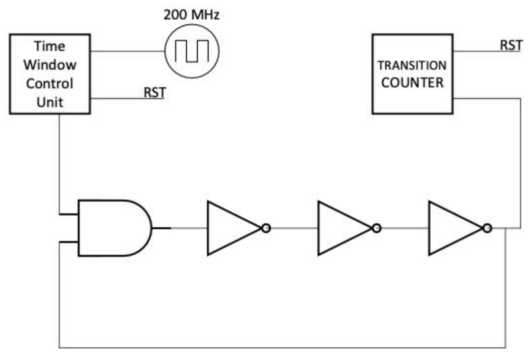


Fig.2 Ring oscillator with a time window control unit and a transition counter.

- **Aging Effect**:
  - Causes delay through gradual degradation.
  - Factors: High temperature, voltage stress.
- **Path Delay Trojan**:
  - Adds delays along rare paths in a circuit.
  - Fault is triggered if delay exceeds clock period.

harder to detect, especially through conventional side-channel and testing techniques.

Y. Yang, J. Ye, X. Li, Y. Han, H. Li, and Y. Hu, "Implementation of Parametric Hardware Trojan in FPGA," *2019 IEEE International Test Conference in Asia (ITC-Asia)*, Tokyo, Japan, 2019, pp. 37-42, doi: 10.1109/ITC-Asia.2019.00020.

Attack Scenario and method

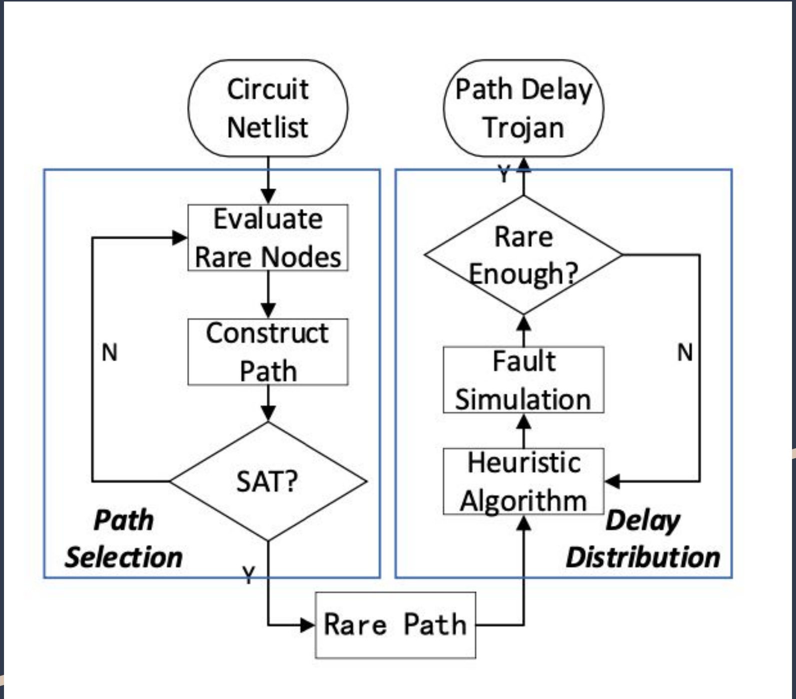


Fig.1 Flow chart of path delay Trojan design.

- **Trojan Implementation**:
  - Increases delay by creating **aging circuits** with ring oscillators.
  - Targets rare paths to increase path delay subtly.
- **Process**:
  - **Select rare path** for Trojan insertion.
  - **Construct aging circuits** using ring oscillators.
  - **Run aging at high temperature/voltage** to introduce delays.
- **Advantages**:
  - No additional circuits; hard to detect with traditional methods.
  - Effective in stealthy attacks on FPGA systems.

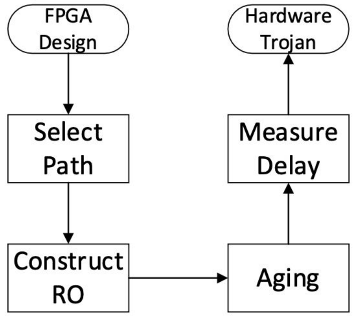


Fig.4 Implementation flow.

Y. Yang, J. Ye, X. Li, Y. Han, H. Li, and Y. Hu, "Implementation of Parametric Hardware Trojan in FPGA," *2019 IEEE International Test Conference in Asia (ITC-Asia)*, Tokyo, Japan, 2019, pp. 37-42, doi: 10.1109/ITC-Asia.2019.00020.

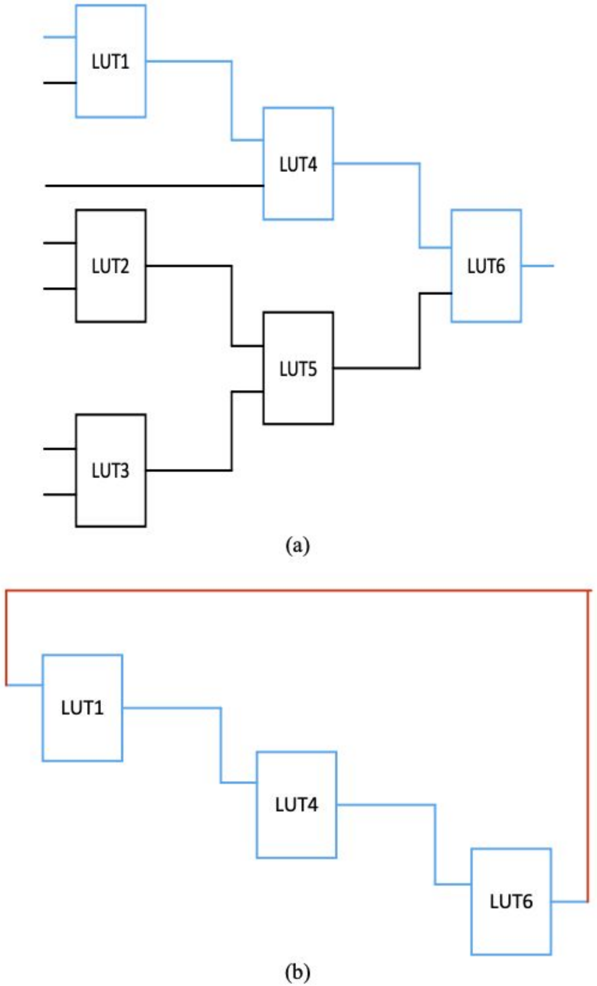
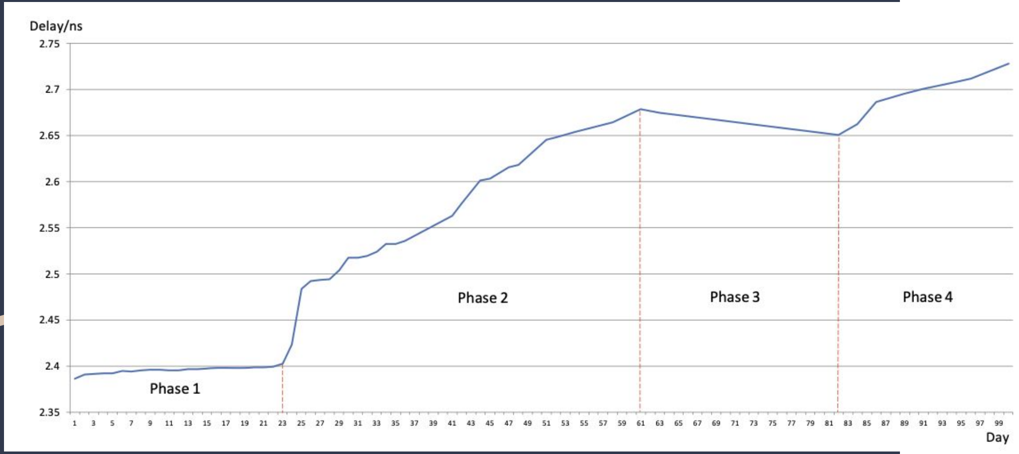


Fig.5 Example of constructing RO for the original circuit

Results

- **Experimental Setup**:
  - Xilinx ZC702 FPGA board
  - **Aging Circuit**: Ring oscillator constructed using target LUTs
  - **Temperature and Voltage Control**: Heating cabinet and adjustable supply voltage
- **Phased Aging Experiment**:
  - **Phase 1**: 60°C with normal voltage – minimal delay increase
  - **Phase 2**: 62°C and 1.6V – significant delay increase
  - **Phase 3**: Aging circuit off – partial recovery observed
  - **Phase 4**: Aging resumed – delay continued to increase

Key Findings:
  - Delay increased monotonically over time, with limited recovery
  - Aging effects can be controlled to implement hardware Trojans effectively



Detection Techniques

1. Golden-Free Machine Learning Detection
2. Thermal Imaging

Golden-Free Hardware Detection

Threat Model Assumed:
Foundry cannot be trusted

What is “Golden-Free”？

- Trusted ICs, free of malicious modifications
- Traditional way: compare chips to golden reference
- Hard and costly to obtain the reference

So, what to do?

Unsupervised clustering techniques

Analyze patterns using side-channel data from RO

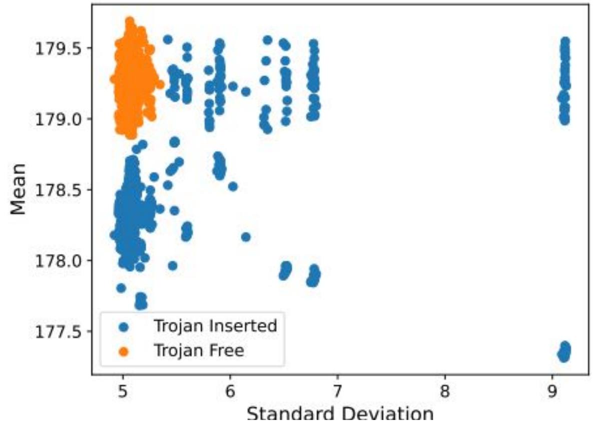
Methodology

K-means: minimized the sum of squared distances between data points and centroids to divide the data

AGNES: hierarchical clustering to identify structural relationships among samples

1. **Data Collection**: Trojan-free & Trojan-inserted ICs, 7001 samples each
2. **Preprocessing**: noise reduction, feature extraction, normalization, irrelevant features removed
3. **Feature Extraction**: central tendencies and variation of RO frequency mean (central tendency of distribution) & standard deviation (for frequency dispersion)
4. **Clustering Models**: K-means and Agglomerative Nesting for unsupervised clustering (meaning they don't rely on labeled data to learn the structure or identify clusters)
5. **Parameter Tuning**: Optimal clusters via elbow plot & dendrogram
6. **Evaluation Metrics**: Accuracy, Precision, F1 score, AUC, FPR (labels are applied afterward to assess the performance of the model by comparing the clusters formed to the actual known labels)

Ghimire et al., "FPGA Hardware Trojan Detection: Golden-Free ML Approach," NAECON, 2023.



- the AGNES model separates Trojan-infected and Trojan-free samples

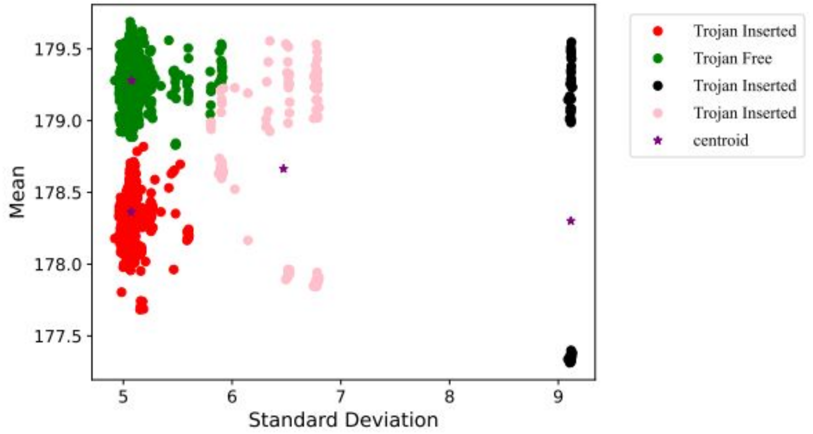
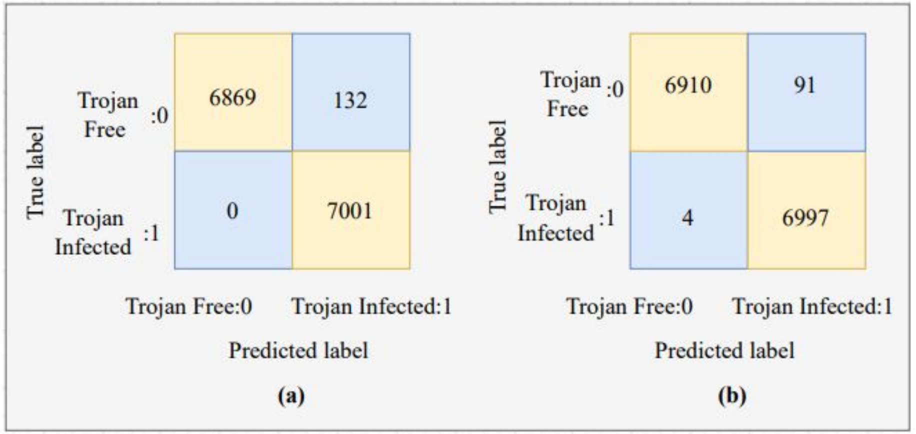


Fig. 6: Scatter plot showing clusters with Kmeans model



(a)

True label \ Predicted label	Trojan Free:0	Trojan Infected:1
Trojan Free :0	6869	132
Trojan Infected :1	0	7001

(b)

True label \ Predicted label	Trojan Free:0	Trojan Infected:1
Trojan Free :0	6910	91
Trojan Infected :1	4	6997

Fig. 7: Confusion Matrix from clustering models(a) AGNES, (b) Kmeans.

Results

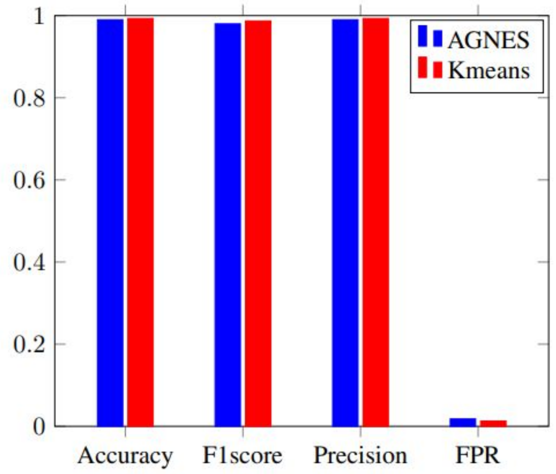


Fig. 10: Comparison of performance of AGNES and Kmeans clustering models

Architecture: RON with 8 FPGA boards, each divided into 4 areas

Cluster Analysis:

- K-means (Centroid-based), AGNES (Hierarchical)
- Cluster size: 4 for K-means, 2 for AGNES (using dendrogram)

Performance Metrics:

- High AUC (0.99), K-means accuracy (0.993), AGNES accuracy (0.991)
- Precision: K-means (0.987), AGNES (0.981)
- FPR: K-means (0.013), AGNES (0.018)

Results Visualization:

- ROC Curves confirm strong discriminative ability
- Confusion matrices show effective Trojan detection

Pazira et al., "Hardware Trojan Detection Using Thermal Imaging in FPGAs," ICSPIS, 2021.

Thermal Imaging

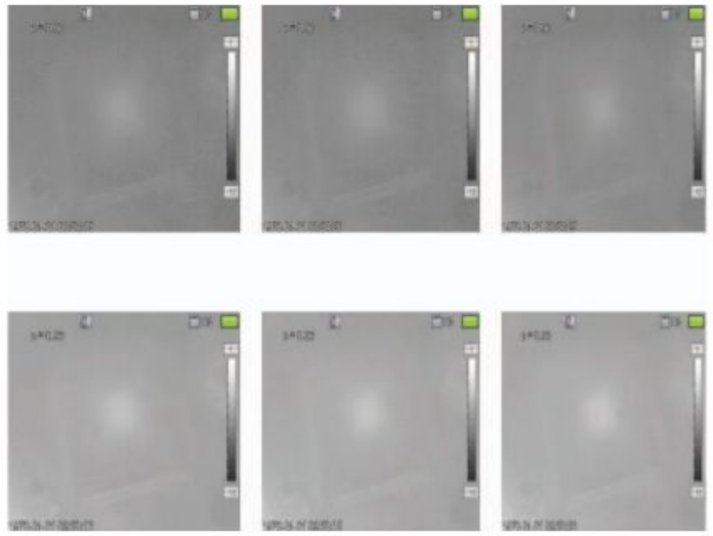


Figure 1. Thermal images in the first line are taken within 15 seconds of Basic_rsa benchmark circuit free of Trojan and the second line contains images within 15 seconds of Basic_rsa with trojan.

T4 FLIR thermal camera

‘dataset consists of 40 different experiments on Spartan-3E FPGA with different circuits, and each experiment included 12 runtime images taken in 55 seconds’

Pazira et al., "Hardware Trojan Detection Using Thermal Imaging in FPGAs," ICSPIS, 2021.

Methodology

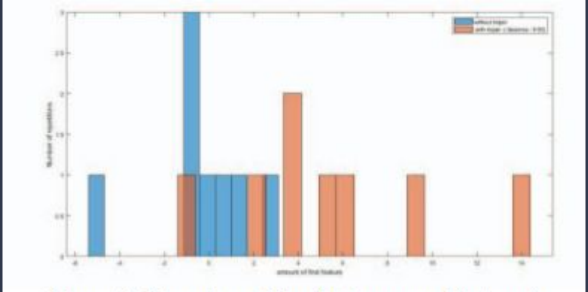


Figure 2. The value of the first proposed feature in with/without HT circuit

Features Used:

- **Thermal Variation**: Difference between sequential thermal images
- **Pixel Threshold**: Count of pixels exceeding 0.2°C in thermal images

Normalization: Equation applied based on pixel count
Framework:

- Thermal data processing and feature extraction
- 2D visualization of Trojan vs. Trojan-free circuits

Base Method: Uses largest eigenvalues from thermal image matrices
Combined Method: Integrates proposed and base methods for robust classification

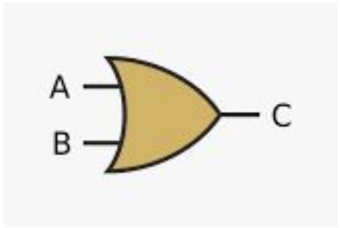


Figure 3 demonstrates the overall framework of the proposed method.

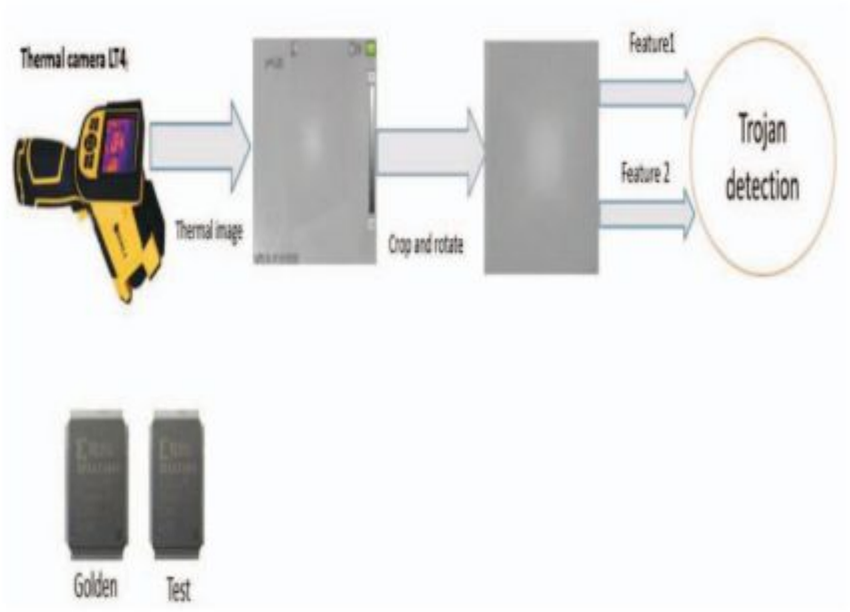


Figure 3. Framework of proposed method

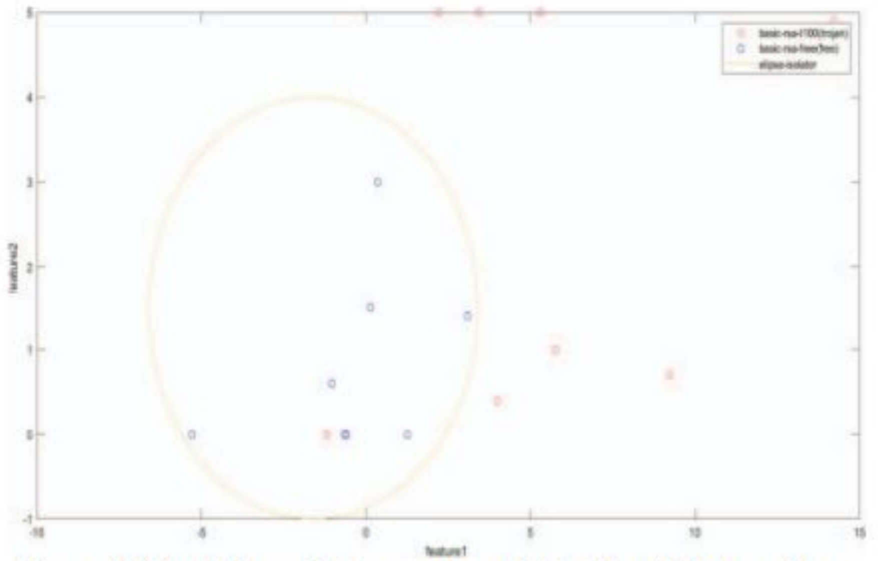


Figure 4. Oval shape that separates with/without Trojan chips in the 2D space of proposed features.

least squares error method

Results

Pazira et al., "Hardware Trojan Detection Using Thermal Imaging in FPGAs," ICSPIS, 2021.

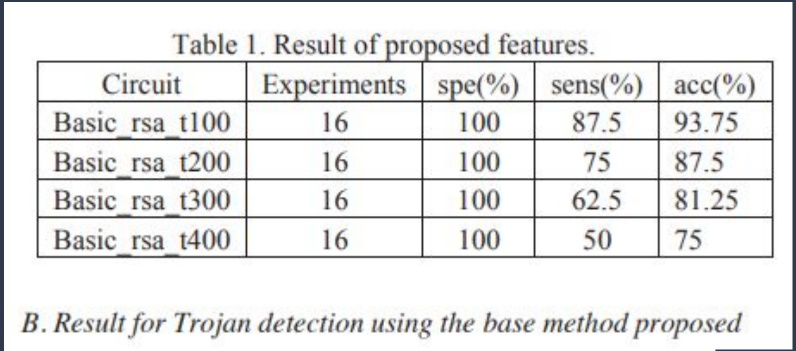


Table 1. Result of proposed features.

Circuit	Experiments	spe(%)	sens(%)	acc(%)
Basic_rsa_t100	16	100	87.5	93.75
Basic_rsa_t200	16	100	75	87.5
Basic_rsa_t300	16	100	62.5	81.25
Basic_rsa_t400	16	100	50	75

B. Result for Trojan detection using the base method proposed

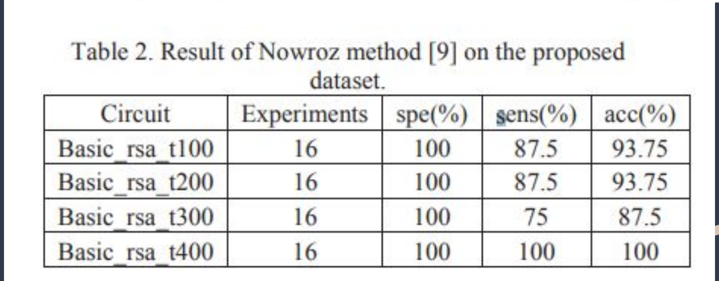


Table 2. Result of Nowroz method [9] on the proposed dataset.

Circuit	Experiments	spe(%)	sens(%)	acc(%)
Basic_rsa_t100	16	100	87.5	93.75
Basic_rsa_t200	16	100	87.5	93.75
Basic_rsa_t300	16	100	75	87.5
Basic_rsa_t400	16	100	100	100

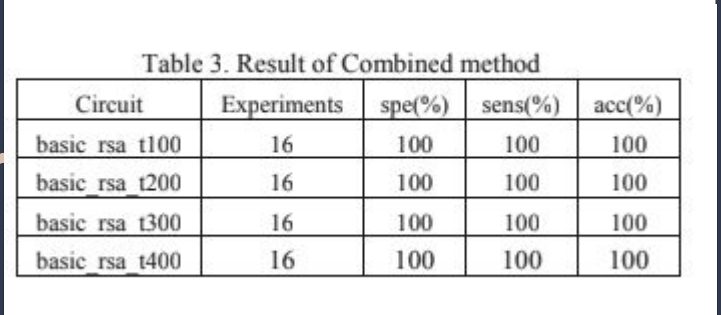


Table 3. Result of Combined method

Circuit	Experiments	spe(%)	sens(%)	acc(%)
basic_rsa_t100	16	100	100	100
basic_rsa_t200	16	100	100	100
basic_rsa_t300	16	100	100	100
basic_rsa_t400	16	100	100	100

Dataset: 40 experiments with 4 Basic_RSA circuits (Trust-hub.org)

Evaluation Metrics: Accuracy, Sensitivity, Specificity

Proposed Method Results:

- **Basic_RSA circuits**: Up to 93.75% accuracy

Comparison with Baseline:

- Similar accuracy in some cases but improved sensitivity

Combined Detection Approach: Achieved 100% accuracy across circuits
Conclusion:

- Low-cost, effective method using IR thermal images
- Combines average variation & pixel counting for detection
- Optimized for budget-conscious applications
- Future work: Enhanced segmentation and multi-algorithm voting

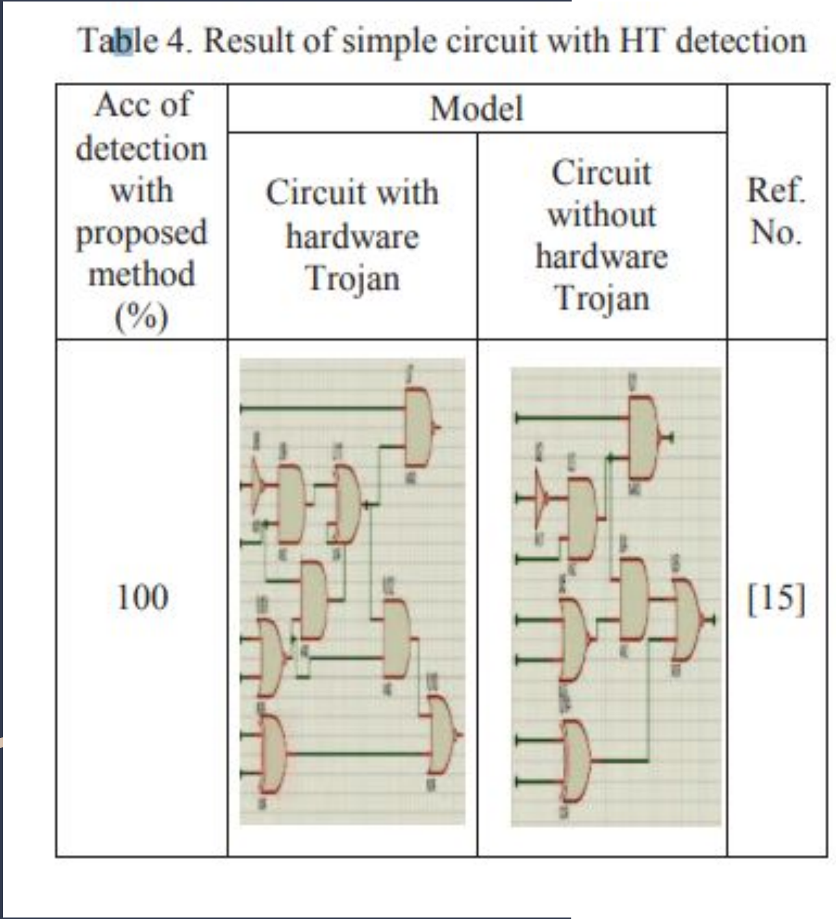
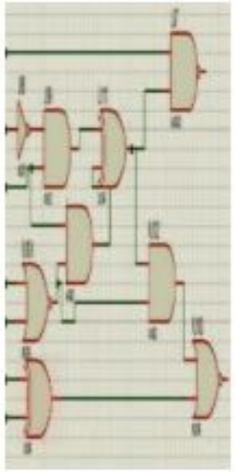
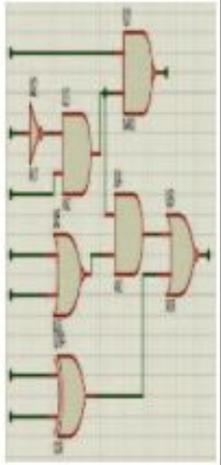


Table 4. Result of simple circuit with HT detection

Acc of detection with proposed method (%)	Model		Ref. No.
	Circuit with hardware Trojan	Circuit without hardware Trojan	
100			[15]

Security Countermeasures and Best Practices

Implement Access Controls: Limit access to sensitive FPGA design and configuration files.

Golden Model Comparison: Use golden-free or trusted baseline models to detect anomalies.

Physical and Environmental Monitoring: Employ side-channel analysis (e.g., thermal imaging, power monitoring).

Diversity in Design: Rotate and modify logic designs to minimize predictability and Trojan insertion points.

Secure Supply Chain: Work with verified suppliers to mitigate risks of tampered components.

Post-Silicon Testing: Conduct thorough post-manufacturing tests for detecting potential Trojans.

User Education: Train staff to recognize and report anomalies in FPGA usage.

Future Research Directions

Advanced Detection Methods: Machine learning models for complex Trojan detection without golden references.

Enhanced Side-Channel Analysis: Focus on non-invasive methods like RF emissions and thermal patterns.

Focus on AI and ML Integration: Leverage AI for real-time detection and adaptive security measures.

Exploring Quantum Security: Investigate quantum methods for enhanced FPGA security and detection.

Citations

A. Murtaza, M. A. Pasha, S. Masud, M. Y. Qadri and A. Basit, "FPGA Based Intelligent Hardware Trojan Design and its SoC Implementation," 2023 30th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Istanbul, Turkiye, 2023, pp. 1-4, doi: 10.1109/ICECS58634.2023.10382763. keywords: {Training;Transmitters;Integrated circuit interconnections;Hardware;System-on-chip;Encryption;Trojan horses;RISC-V;Wishbone;Hardware Trojan;IP Core;FPGA},

Y. Nozaki, S. Takemoto, Y. Ikezaki and M. Yoshikawa, "LUT oriented Hardware Trojan for FPGA based AI Module," 2020 6th International Conference on Applied System Innovation (ICASI), Taitung, Taiwan, 2020, pp. 46-49, doi: 10.1109/ICASI49664.2020.9426247. keywords: {Technological innovation;Logic gates;Hardware;Large scale integration;Table lookup;Security;Trojan horses;hardware Trojan;AI security;FPGA;and hardware security},

Y. Yang, J. Ye, X. Li, Y. Han, H. Li and Y. Hu, "Implementation of Parametric Hardware Trojan in FPGA," 2019 IEEE International Test Conference in Asia (ITC-Asia), Tokyo, Japan, 2019, pp. 37-42, doi: 10.1109/ITC-Asia.2019.00020. keywords: {Trojan horses;Hardware;Field programmable gate arrays;Delays;Aging;Ring oscillators;Logic gates;FPGA;Hardware Trojan;Delay},

A. Ghimire, F. Amsaad, T. Hossain, T. Hoque and A. Sherif, "FPGA Hardware Trojan Detection: Golden-Free Machine Learning Approach," NAECON 2023 - IEEE National Aerospace and Electronics Conference, Dayton, OH, USA, 2023, pp. 181-186, doi: 10.1109/NAECON58068.2023.10365812. keywords: {Integrated circuits;Semiconductor device modeling;Supply chains;Training data;Machine learning;Hardware;System-on-chip;IC Security;Clustering;Trojan;FPGA;side-channel;unsupervised},

M. Pazira, Y. Baleghi and A. Akbari, "Hardware Trojan Detection Using Thermal Imaging in FPGAs with Combined Features," 2021 7th International Conference on Signal Processing and Intelligent Systems (ICSPIS), Tehran, Iran, Islamic Republic of, 2021, pp. 1-5, doi: 10.1109/ICSPIS54653.2021.9729357. keywords: {Vibrations;Signal processing algorithms;Signal processing;Feature extraction;Cameras;Hardware;Thermal analysis;Hardware Trojan detection;side channel analysis;thermal image analysis;FPGA},

Q&A