



Detecting HLS Hardware Trojans using Random Forest Classifier

Authors: Anirban Sengupta and Nabendu Bhui

IEEE Design & Test, 2026

Detecting HLS Hardware Trojans using Random Forest Classifier

- **Introduction**

- Various modern complex electronic and multimedia system/devices heavily depend on secure computing processor (system-on-chips (SoC)) that require computer-aided design (CAD) flow at different abstraction levels of IP designs.
- High level synthesis (HLS) generated IP designs are essential in different electronic and multimedia systems, as HLS is an industry standard CAD flow to design custom IPs.
- Some examples of HLS-generated hardware IP designs that are being widely used in modern complex systems are discrete cosine transform (DCT), convolutional layer (CNN), finite impulse response (FIR), sharpening filter (SF), 4-point DCT, 8-point DCT, blur filter, discrete wavelet transform (DWT), vertical embossment filter (VE), horizontal embossment filter (HE), cardiac pacemaker (CP) etc.
- Although HLS flow is highly beneficial for the productivity of IP design process, this abstraction opens a path for malicious attackers/adversaries to carry out backdoor Trojan attacks.
- A backdoor hardware Trojan generally refers to a harmful/malicious alteration of the IP design intended to obtain confidential information or change system functionality by causing system malfunctions or performance degradation attacks or denial-of-service attacks etc.
- Hardware Trojans may be injected directly by an untrustworthy entity in a third-party CAD tool (such as HLS tool) at any stage of the integrated circuit design process. It results into malicious/compromised register transfer level (RTL) design files of the IP design.

• Introduction : Novel Contributions

- However, several prior works have resolutely established major security vulnerabilities that HLS design process exposes.
- These vulnerabilities sanction malicious Trojan to be furtively injected during HLS design phases such as scheduling, mux-interconnect design stage, RTL datapath etc, resulting into compromised IP designs.
- Therefore, identification/detection of HLS hardware Trojans is very essential during the IP core design stage of the global design supply chain.

➤ Novel Contributions

- Our paper presents novel technique for detecting HLS hardware Trojans (HLS-HT) using random forest classifier-based machine learning (ML) model.
- The proposed detection framework is capable of detecting various types of state-of-the-art HLS-HT including performance degradation hardware Trojan (**PD-HT**), denial of service hardware Trojan (**DoS-HT**), battery exhaustion hardware Trojan (**BE-HT**), downgrade attack hardware Trojan (**DA-HT**), functional hardware Trojan (**F-HT**) and Time-bomb hardware Trojan (**TB-HT**).
- A novel feature extraction block is proposed and designed, which extracts the several crucial unique features from the register transfer level (RTL) description of hardware IP designs (VHDL code).
- The proposed approach achieves high detection accuracy with zero false negatives for specific HLS-HTs considered here, ensuring sturdy IP design security.

- HLS Hardware Trojans (HLS-HTs)

➤ **Fig. 1** illustrates six specific class of state-of-the-art (SOTA) HLS Trojans along with its insertion/attack sites.

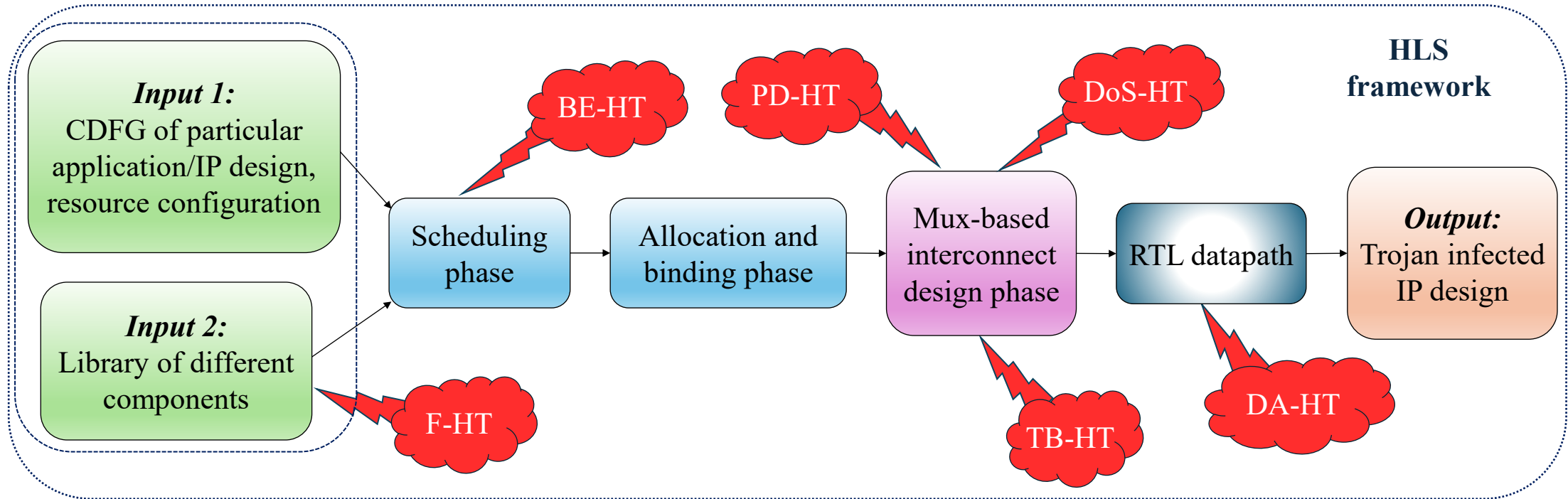


Fig. 1. Different HLS Trojans in the HLS framework

• HLS Hardware Trojans (HLS-HTs) : Details

➤ Various state-of-the-art (SOTA) HLS-HTs:

▪ Functional hardware Trojan (**F-HT**) [5]

Leverages a 2:1 Mux and inverter to change the functionality of used functional units.

- *Insertion stage* → HLS library
- *Triggering mechanism* → External signal, Finite State Machine (FSM) counter
- *Payload* → Compromising functionality/data damage

▪ Denial of service hardware Trojan (**DoS-HT**) [4]

Leverages tri-state buffer (TSB) to induce high impedance state.

- *Insertion stage* → Mux-interconnect design
- *Triggering mechanism* → Comparator based trigger
- *Payload* → Denial-of-service

[4] A. Sengupta, *et. al.*, M-HLS: Malevolent High-Level Synthesis for Watermarked Hardware IPs,” *IEEE Embedded Systems Letters*, vol. 16, no. 4, pp. 497-500, Dec. 2024.

[5] A. Sengupta, *et. al.*, TL-HLS: Methodology for Low Cost Hardware Trojan Security Aware Scheduling With Optimal Loop Unrolling Factor During High Level Synthesis, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol.36 (4), pp.655-668, 2017.

• HLS Hardware Trojans (HLS-HTs) : Details

➤ Various SOTA HLS-HTs:

▪ Performance degradation hardware Trojan (**PD-HT**) [4]

Leverages chain/series of inverters to induce excessive delay.

- *Insertion stage* → Mux-interconnect design
- *Triggering mechanism* → Comparator based trigger
- *Payload* → Performance degradation

▪ Battery exhaustion hardware Trojan (**BE-HT**) [3]

Leverages unutilized functional units to induce excess power drainage based on a certain execution count.

- *Insertion stage* → Scheduling/FSM
- *Triggering mechanism* → Counter based trigger
- *Payload* → Battery exhaustion/power drainage

[3] C. Pilato, *et. al.*, “Black-Hat High-Level Synthesis: Myth or Reality?,” *IEEE TVLSI*, vol. 27, no. 4, pp. 913-926, April 2019.

[4] A. Sengupta, *et. al.*, M-HLS: Malevolent High-Level Synthesis for Watermarked Hardware IPs,” *IEEE Embedded Systems Letters*, vol. 16, no. 4, pp. 497-500, Dec. 2024.

• HLS Hardware Trojans (HLS-HTs) : Details

➤ Various SOTA HLS-HTs:

- Downgrade attack hardware Trojan (**DA-HT**) [3]
Leverages a 2:1 Mux to implement downgrade attack.
 - *Insertion stage* → Datapath
 - *Triggering mechanism* → Input sequence detector
 - *Payload* → Downgrade attack/compromised security

- Time-bomb hardware Trojan (**TB-HT**) [6]
Leverages chain/series of TSBs to induce excessive performance delay.
 - *Insertion stage* → Mux-interconnect design
 - *Triggering mechanism* → Comparator and counter based trigger
 - *Payload* → Performance degradation

[3] C. Pilato, *et. al.*, “Black-Hat High-Level Synthesis: Myth or Reality?,” *IEEE TVLSI*, vol. 27, no. 4, pp. 913-926, April 2019.

[6] A. Sengupta, N. Bhui “Time-Bomb HLS Trojan for Performance Degradation Payload,” *IEEE Design & Test*, vol. 42(5), pp. 83-92, 2025

• HLS Hardware Trojans (HLS-HTs) : Summary

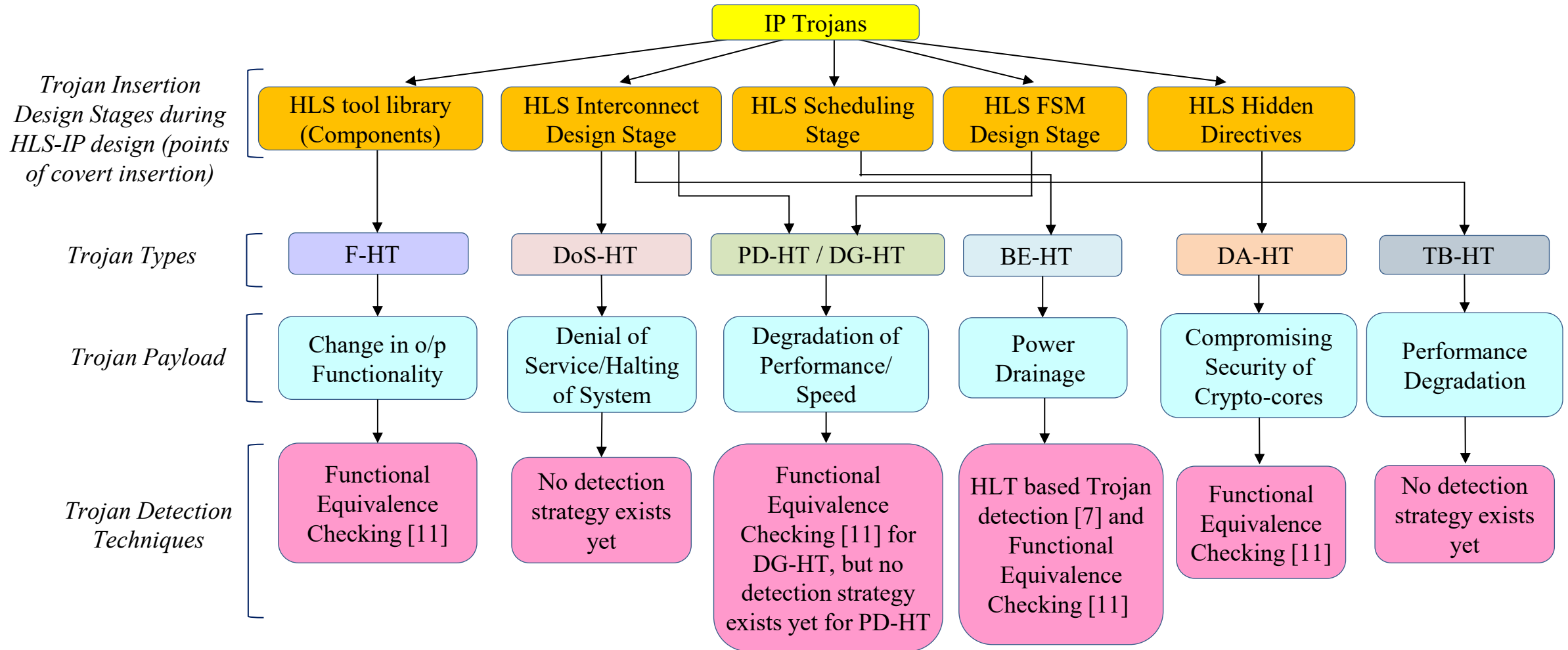


Fig. 2. HLS-IP based Trojan Classification and its Detective Countermeasures

➤ The proposed detection framework is capable of detecting these types of state-of-the-art HLS-HTs.

• Related Works

TABLE 1
RELATED WORKS

Sl. No.	Existing Works	Methodology	Remarks
1.	A. Sengupta et al., [5] (2017)	Dual modular redundant (DMR) based trojan detection technique that solely performs detection/identification of F-HT.	[5] cannot identify PD-HT, DoS-HT, BE-HT, DA-HT, and TB-HT.
2.	M. Abderehman et al., [11] (2022)	C to RTL equivalence checking demands Trojan-infected C-code from which an FSM with datapath (FSMD) can be generated.	HLS Trojans such as PD-HT, DoS-HT and TB-HT are not available as Trojan infected C-code, as the Trojan insertion was performed in mux-interconnect design stage of HLS.
3.	A. Sengupta et al., [12] (2023)	Triple modular redundant (TMR) based HLS Trojan resistant technique facilitates the detection and isolation of solely F-HT in the produced RTL datapath.	[12] does not offer detection/identification of PD-HT, DoS-HT, BE-HT, DA-HT, and TB-HT.
4.	R. Yasaei et al., [9] (2025)	Uses graph neural networks (GNNs) to detect RTL inserted HTs. However this technique is language dependent as it uses Pyverilog toolkit to parse and analyze the Trojan-infected Verilog code.	[9] can only detect RTL-HTs, as it trains its model on simple non-HLS benchmarks such as AES, DES etc. It does not apply for complex HLS designs.
5.	M. Rathor et al., [7] (2024)	High-level transformation (HLT) based Trojan detection technique uses HLTs to analyze/assess operation counts. It is used to counter BE-HT attack.	[7] does not work against other five type of HLS Trojans. This is due to the fact that these HLS Trojans do not depend on adding fake operations to affect the Trojan payload.
6.	K. Hasegawa et al., [8] (2017)	Machine Learning (ML) detection technique [8] is used for identifying RTL Trojans or HTs at netlist level.	However, these works do not focus on HLS-aided Trojan detection, unlike the proposed approach.

➤ Therefore, there has been no prior work which offer detection/identification of all these SOTA HLS-HTs (F-HT, DoS-HT, PD-HT, BE-HT, DA-HT, and TB-HT).

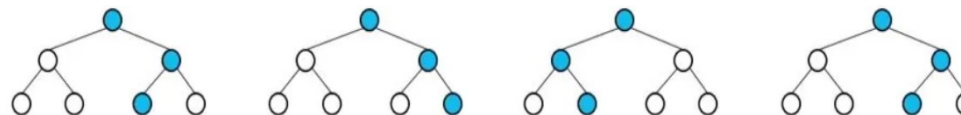
• Threat Model

➤ Threat Model handled in the Paper

- The abstract and automated characteristics of HLS tools used in HLS-based hardware IP design process creates a path for attackers/adversaries to insert Trojan/malicious logic, which poses security vulnerabilities for the IP designs.
- The HLS tool design flow presents significant security vulnerabilities due to two primary factors [3]-[5]:
 - (i) Initially, HLS design process may introduce security vulnerabilities, enabling the insertion of hardware Trojans/malicious logics during IP design, which ultimately compromises the quality of the produced IP or results in substandard IP components in the global market,
 - (ii) Secondly, the Trojan inserted HLS framework may also be part of national-level toolkit, where tools designed/developed in one nation and can be utilized in different nation to build the SoCs.
- These vulnerable HLS tools may contain backdoor injected Trojan or malicious logic secretly embedded in different phases of HLS design process such as HLS library stage, resource allocation phase, scheduling/FSM phase, interconnect design stage, and datapath or controller synthesis phase [3]-[6].

• Motivation of using Random Forest ML Classifier

- We have taken the advantage of Random Forest (RF) classifier-based ML framework to identify/detect/classify the presence of HLS-Trojan in the register transfer level (RTL) description of hardware IP designs.
- The RF classifier is a popular ensemble learning technique that leverages the strengths of various decision trees to enhance predictive accuracy/precision and model stability. RF can handle both numerical and categorical data efficiently.
- It can also detect nonlinear patterns and interactions among features without significant data transformation. It is well-suited for classification tasks because of its robustness, ability to model complex data patterns, and lower chance of overfitting.
- In contrast to algorithms like support vector machines (SVM), which rely on kernel functions and demand careful feature scaling, RF is more user-friendly and scalable. It is generally more tolerant to hyperparameter tunings compared to models like SVM, neural networks, or deep learning models, which often require precise tuning, and they are sensitive to proper scaling of features.
- In contrast to deep learning models that require significant resources and are complicated to fine-tune, RF attains high accuracy with a reasonable computational cost.
- Considering these benefits, the RF classifier serves as a practical and effective option for HLS hardware Trojan detection/classification, which provides robust predictive performance with limited preprocessing and tuning effort.



• The Proposed Methodology

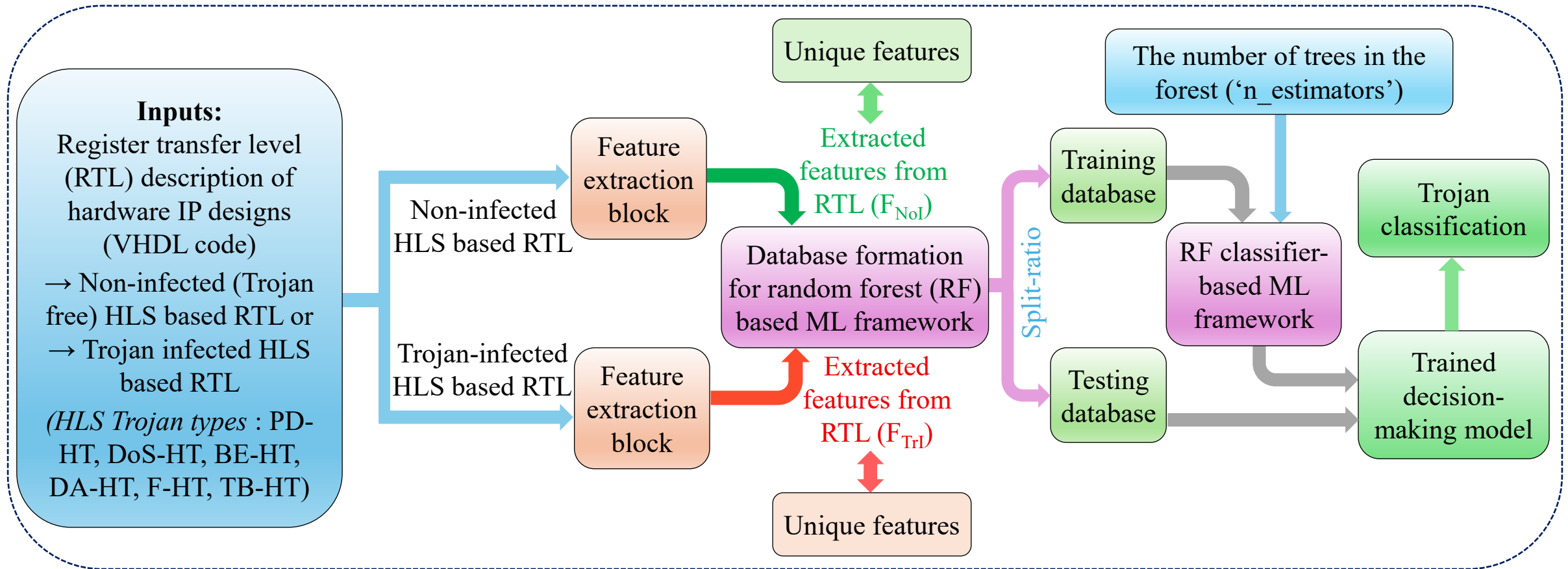


Fig. 3. Proposed random forest (RF) classifier based Trojan detection framework

• The Proposed Methodology (Contd.)

➤ Dataset used:

- The proposed HLS-Trojan database comprises of multiple instances related to various IP designs.
- The specific details of the IP designs along with their transfer functions and control dataflow graphs (CDFGs) are available publicly in [10].
- The Trojan infected HLS IP benchmarks used are convolutional layer, FIR, MESA, GLRT, IIR, 4-point DCT, Sharpening filter, 8-point DCT, LED filter, Blur filter, DWT, vertical embossment filter, horizontal embossment filter, cardiac pacemaker. These HLS IP benchmarks are standard applications from the domain of DSP, multimedia, and medical systems [4]-[6].
- The respective proposed feature extraction code and generated database is shown in [10].
- Here, HLS Trojan types are PD-HT (**TI1**), DoS-HT (**TI2**), BE-HT (**TI3**), DA-HT (**TI4**), F-HT (**TI5**), and TB-HT (**TI6**) [4], [3], [5], and [6].
- Additionally, the database includes **23 unique extracted features** (F_{NoI} and F_{TrI}) for the IP designs from their corresponding **non-infected** (NoI), and **Trojan infected** (TI1, TI2, TI3, TI4, TI5, and TI6) RTL description.
- Existing HLS Trojans (which are used to generate the database) along with their features are summarized below:

[10] *GitHub Repository*, Available [Online]: https://github.com/NBhui95/HLS_Hardware_Trojans_Detection_using_RF_Classifier.

- The Proposed Methodology (Contd.)

TABLE 2
EXTRACTED FEATURES OF RTL IP DESIGN AND CORRESPONDING HLS TROJANS

Extracted unique features from HLS generated RTL IP design	Non-infected / infected HLS Trojan type
4*1 Mux, 8*1 Mux, 16*1 Mux, 32*1 Mux, 1*2 DeMux, 1*4 DeMux, 1*8 DeMux, 1*16 DeMux, 1*32 DeMux, Latch, Adder, Subtractor, Multiplier, 2*1 Mux	Non-infected
4*1 Mux, 8*1 Mux, 16*1 Mux, 32*1 Mux, 1*2 DeMux, 1*4 DeMux, 1*8 DeMux, 1*16 DeMux, 1*32 DeMux, Latch, Adder, Subtractor, Multiplier, Comparator, 2*1 Mux, Inverter Chain, Mux output acting as input for another Mux, Memory	PD-HT
4*1 Mux, 8*1 Mux, 16*1 Mux, 32*1 Mux, 1*2 DeMux, 1*4 DeMux, 1*8 DeMux, 1*16 DeMux, 1*32 DeMux, Latch, Adder, Subtractor, Multiplier, Comparator, 2*1 Mux, Tri-state buffer, Mux output acting as input for another Mux, Memory, Inverter output acting as input to the TSB	DoS-HT
4*1 Mux, 8*1 Mux, 16*1 Mux, 32*1 Mux, 1*2 DeMux, 1*4 DeMux, 1*8 DeMux, 1*16 DeMux, 1*32 DeMux, Latch, Adder, Subtractor, Multiplier, 2*1 Mux, Up-counter	BE-HT
4*1 Mux, 8*1 Mux, 16*1 Mux, 32*1 Mux, 1*2 DeMux, 1*4 DeMux, 1*8 DeMux, 1*16 DeMux, 1*32 DeMux, Latch, Adder, Subtractor, Multiplier, Comparator, 2*1 Mux	DA-HT
4*1 Mux, 8*1 Mux, 16*1 Mux, 32*1 Mux, 1*2 DeMux, 1*4 DeMux, 1*8 DeMux, 1*16 DeMux, 1*32 DeMux, Latch, Adder, Subtractor, Multiplier, 2*1 Mux, Extra_FU_Input	F-HT
4*1 Mux, 8*1 Mux, 16*1 Mux, 32*1 Mux, 1*2 DeMux, 1*4 DeMux, 1*8 DeMux, 1*16 DeMux, 1*32 DeMux, Latch, Adder, Subtractor, Multiplier, Comparator, 2*1 Mux, TSB chain, Up-counter, Mux output acting as input for another Mux, Memory, Inverter output acting as input to the TSB	TB-HT

• The Proposed Methodology (Contd.)

- Performance degradation hardware Trojan (**PD-HT**) [4]
 - The PD-HT is activated through a comparator unit.
 - Here, Trojan logic gets activated when the comparator input 'i' equals the attacker's preloaded constant 'c'.
 - An attacker adds even number of inverters (inverter chain), causing a reduction in performance due to the increased propagation delay.
 - The proposed ML-based approach can identify PD-HT by extracting the following unique features (which are extracted through the feature extraction code [10]):
 - (a) comparator, (b) extra 2*1 mux, (c) inverter chain,
 - (d) mux output acting as input for another mux, and
 - (e) memory (i.e., pre-loaded constant).

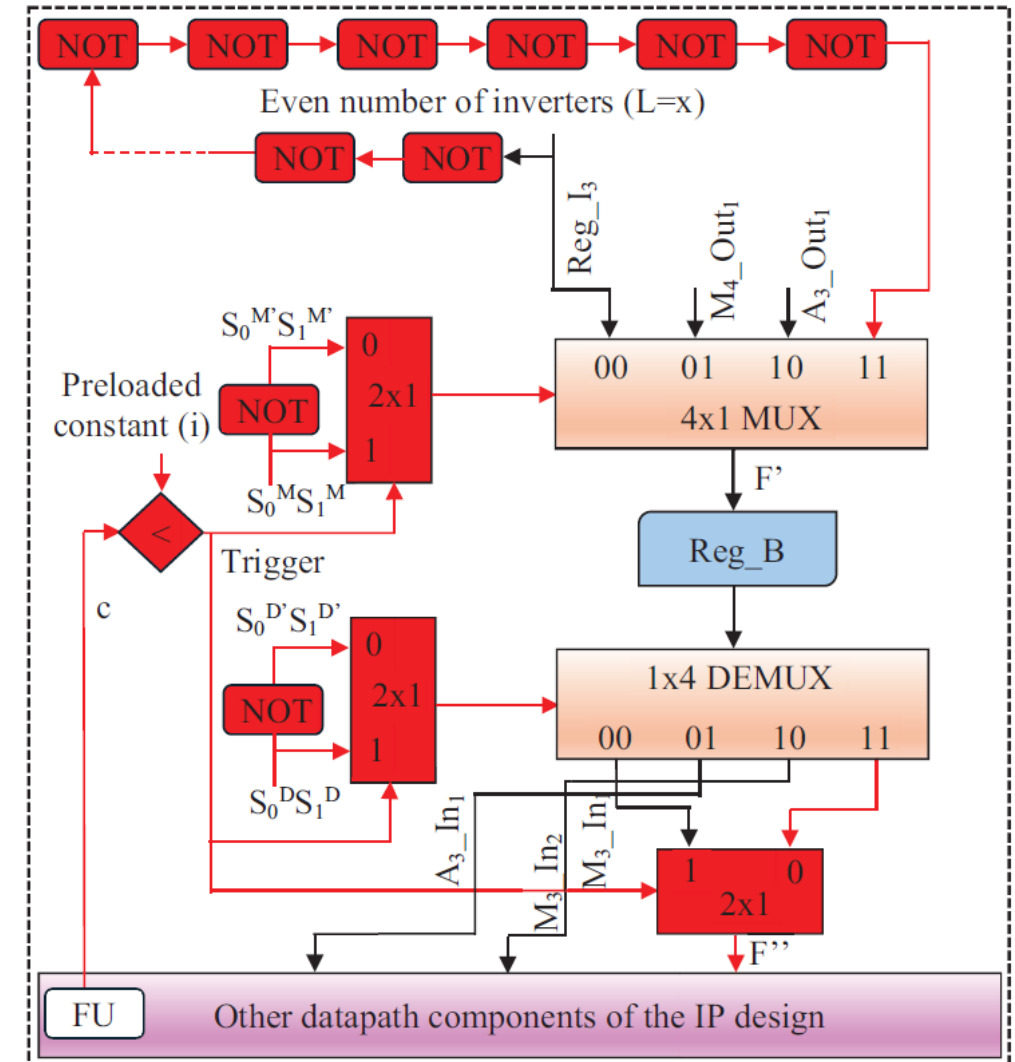


Fig. 4. Performance degradation hardware Trojan (PD-HT)

• The Proposed Methodology (Contd.)

➤ Denial of service hardware Trojan (DoS-HT) [4]

- The DoS-HT is also activated through a comparator unit.
- Trojan logic is activated when the comparator input 'i' equals the attacker's preloaded constant 'c'.
- The attacker/adversary executes a payload that interrupts the operational execution of the hardware IP design.
- The proposed ML-based approach can identify DoS-HT by extracting the following unique features:
 - (a) comparator, (b) extra 2*1 mux, (c) tri-state buffer, (d) mux output acting as input for another mux, (e) memory, and (f) inverter output acting as input to the TSB.

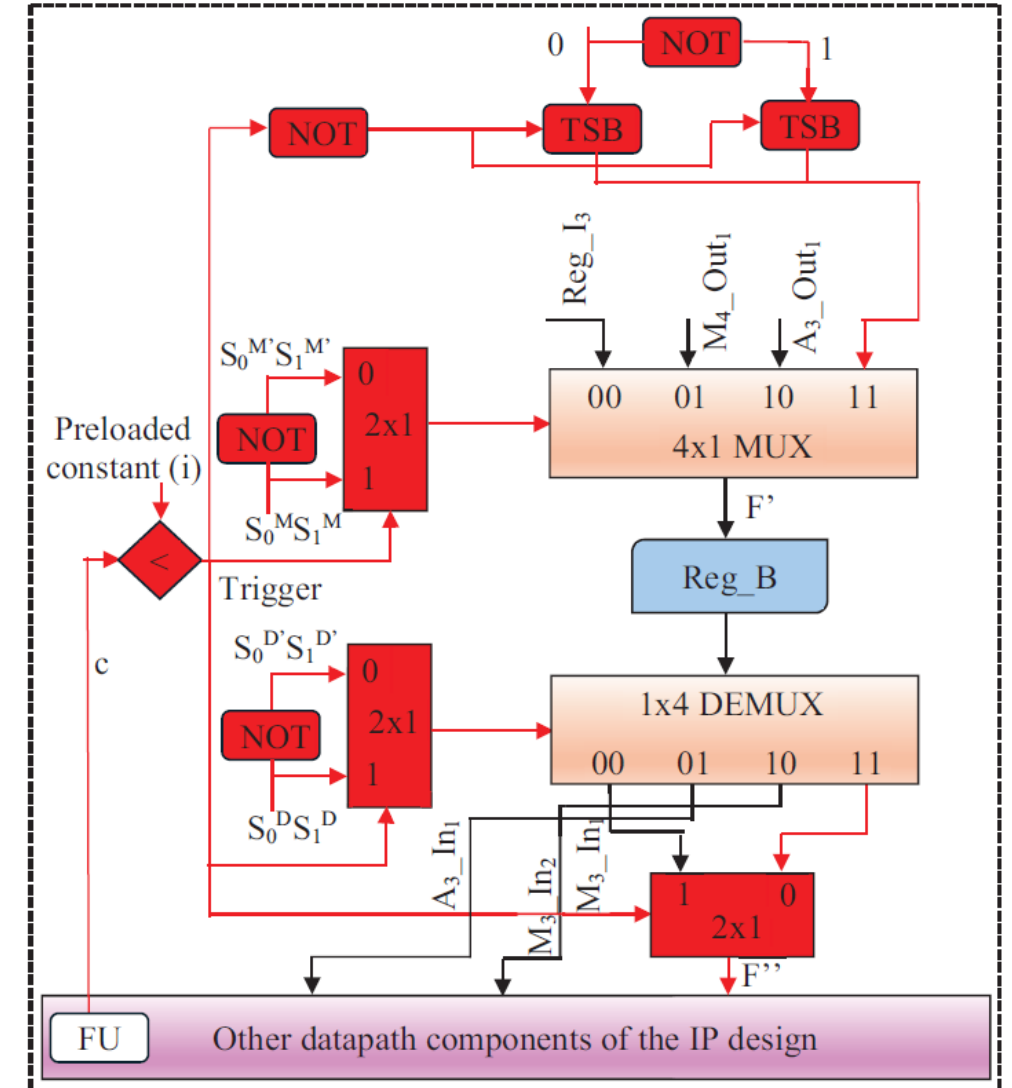


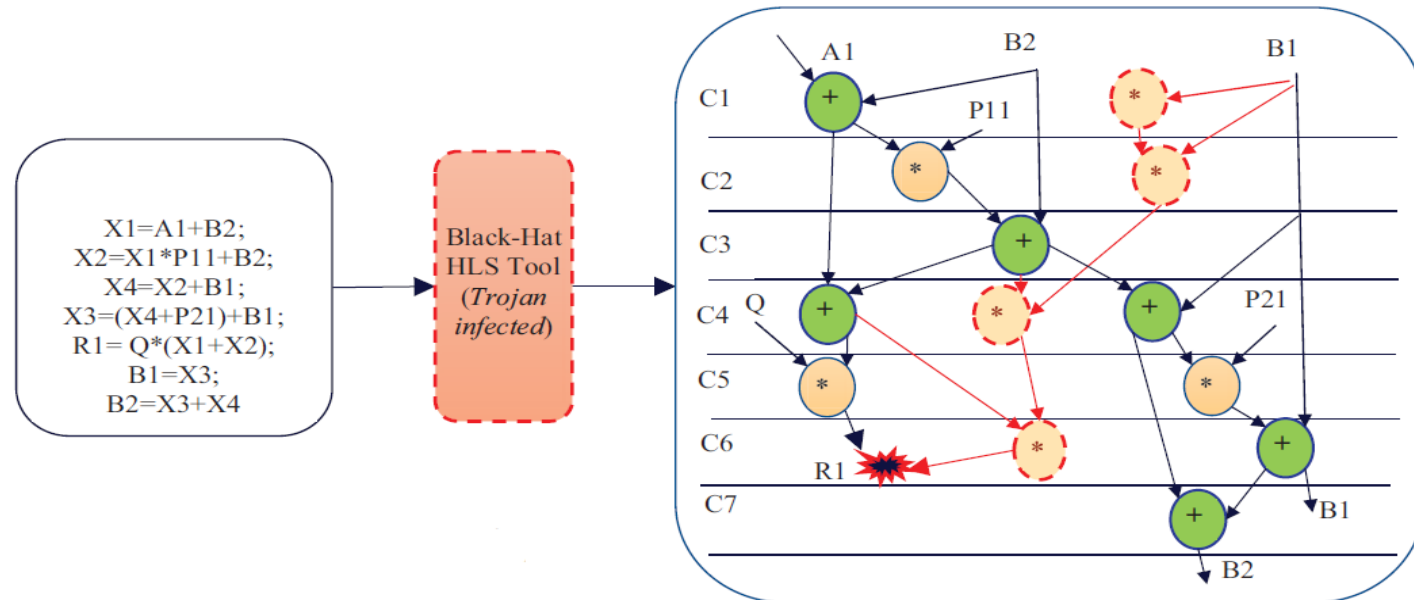
Fig. 5. Denial of Service hardware Trojan (DoS-HT)

• The Proposed Methodology (Contd.)

➤ Battery exhaustion hardware Trojan (**BE-HT**) [3]

- The BE-HT is activated/triggered through an up-counter unit.
- The counter-driven trigger (counter) activates/triggers the Trojan after a predefined specific number of execution.
- The attacker modifies the hardware IP design to boost/enhance the power consumption (by utilizing idle/inactive functional units during a specific control step), speeding up battery discharge.
- The proposed ML-based approach can identify BE-HT by extracting the following unique feature (which are extracted through the feature extraction code [10]): (a) up-counter.

Fig. 6. Illustration of BE-HT attack for IP designs



• The Proposed Methodology (Contd.)

- Downgrade attack hardware Trojan (**DA-HT**) [3]
 - The DA-HT is activated/triggered through an input sequence detector.
 - The attacker undermines the total security of crypto-cores by lowering the rounds of cryptographic algorithm (using a new count value that is smaller than the original).
 - The proposed ML-based approach can identify DA-HT by extracting the following unique features (which are extracted through the feature extraction code [10]):
 - (a) comparator, and (b) extra 2*1 Mux.

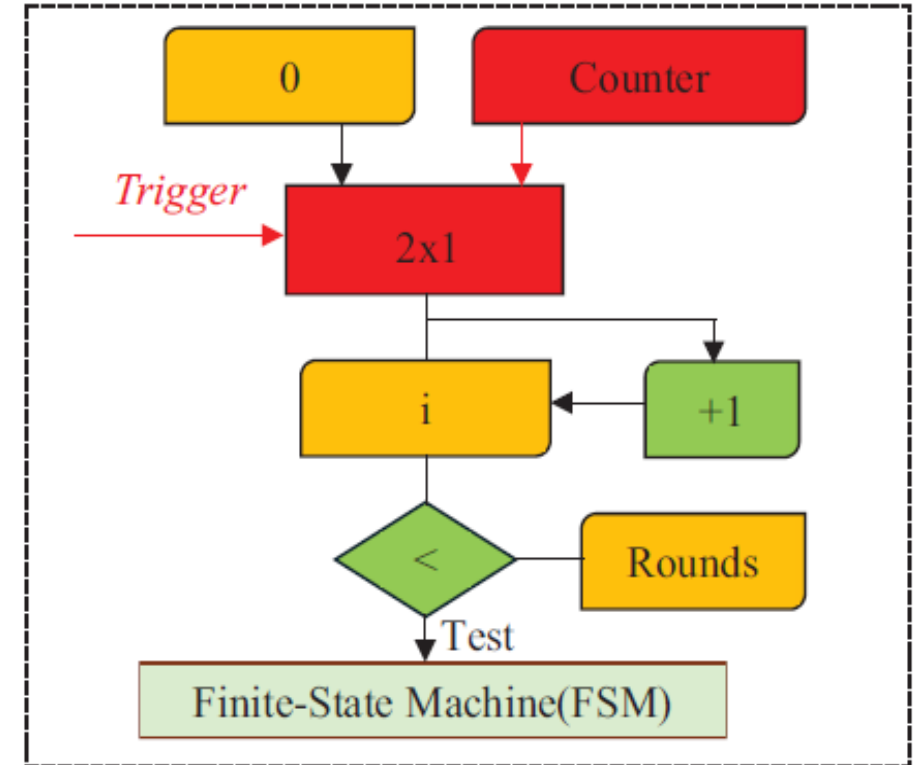


Fig. 7. Downgrade hardware Trojan (DA-HT)

• The Proposed Methodology (Contd.)

➤ Functional hardware Trojan (F-HT) [5]

- The F-HT is activated/triggered through an FSM counters/external signal. It changes the computational output after triggering.
- The proposed ML-based approach can identify F-HT by extracting the following unique features (which are extracted through the feature extraction code [10]):
 - (a) extra_FU_input, and (b) extra 2*1 mux.
- The existence of an additional input port in the functional unit (FU) element signifies a Trojan trigger signal.
- However, an extra input port in a FU component may also signify a true signal. An attacker may take advantage of extra input port for triggering the Trojan.
- Therefore, this extra input port of a FU component is considered as a unique skeptical feature.

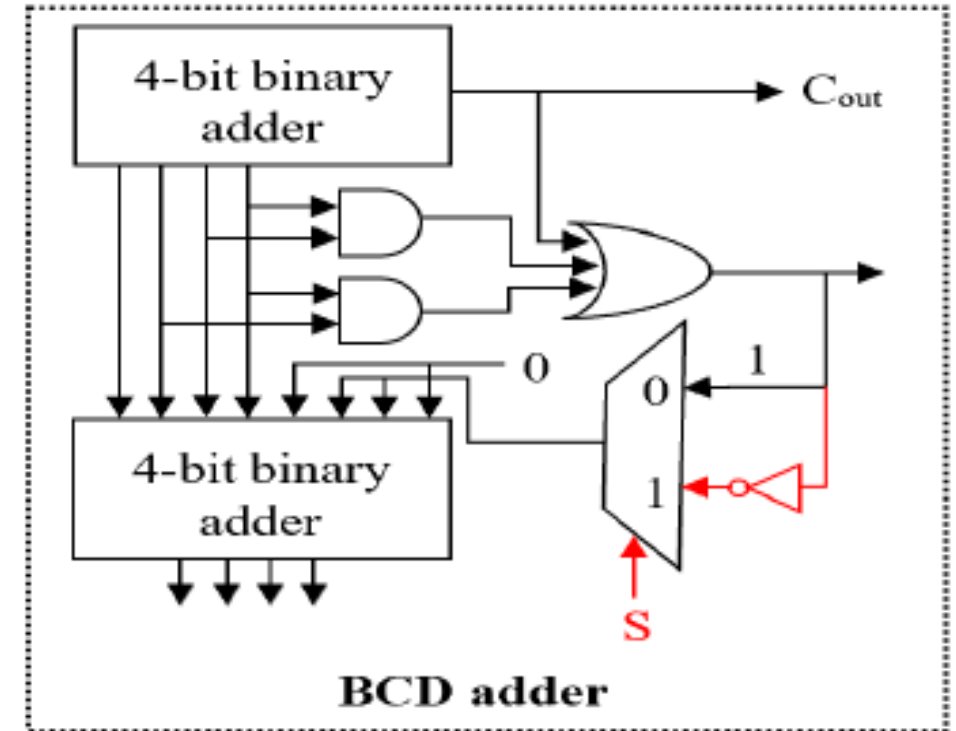


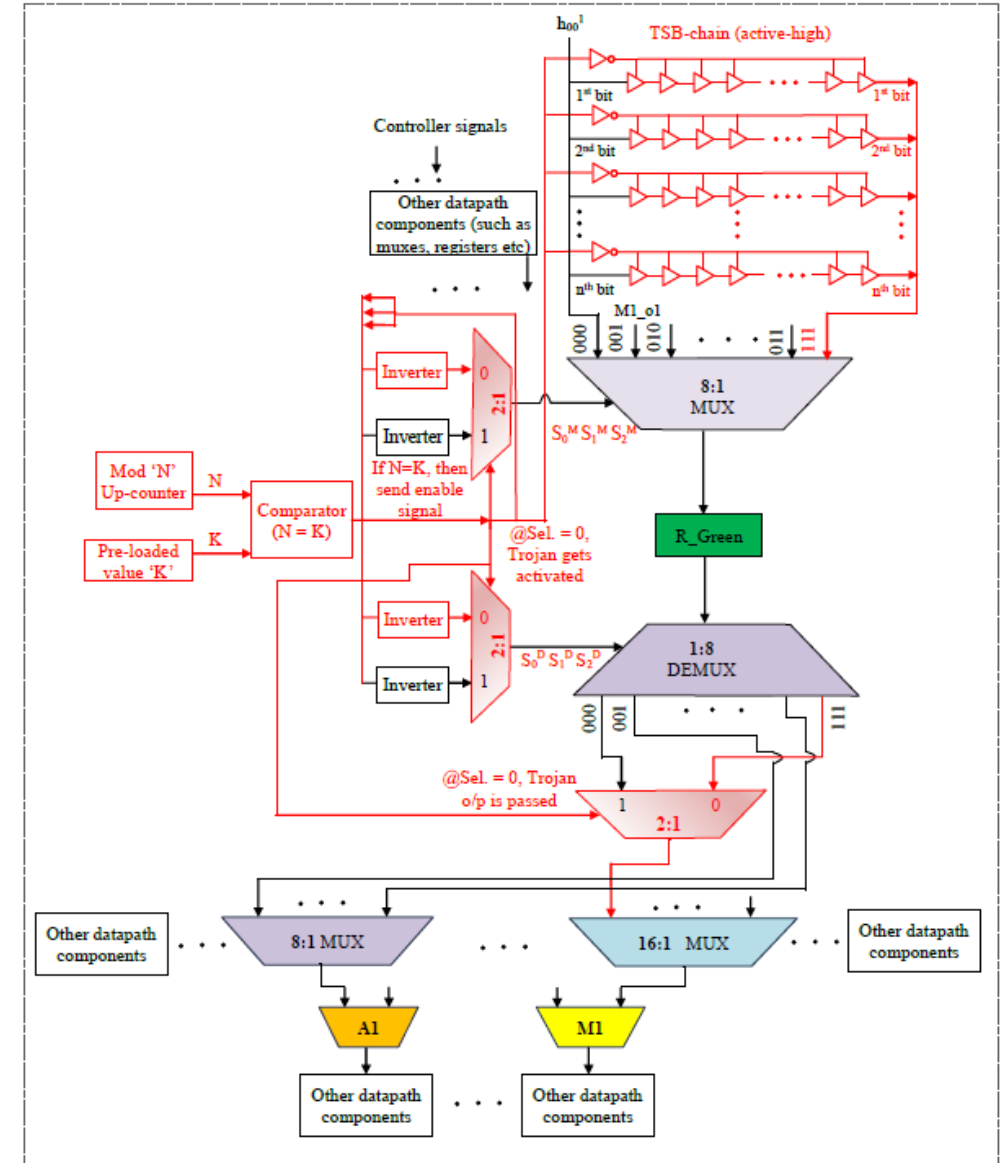
Fig. 8. Trojan example in tool library

• The Proposed Methodology (Contd.)

➤ Time-bomb hardware Trojan (TB-HT) [6]

- The TB-HT is activated/triggered through an integrated counter and comparator unit logic.
- The attacker embeds even number of TSB chains during mux-based interconnect design stage, which results into a huge degradation in performance.
- The proposed ML-based approach can identify TB-HT by extracting the following unique features (which are extracted through the feature extraction code [10]):
 - (a) comparator, (b) extra 2*1 mux, (c) TSB chain, (d) up-counter, (e) mux output acting as input for another mux, (f) memory, and (g) inverter output acting as input to the TSB.

Fig. 9. HLS based time-bomb Trojan inserted convolution filter IP datapath design – partial view



• The Proposed Methodology (Contd.)

➤ Therefore, extracted **23 unique features** are:

*4*1 mux, 8*1 mux, 16*1 mux, 32*1 mux, 1*2 demux, 1*4 demux, 1*8 demux, 1*16 demux, 1*32 demux, latch, adder, subtractor, multiplier, comparator, 2*1 mux, TSB chain, inverter chain, up-counter, Tri-state buffer, mux output acting as input for another mux, extra_FU_input, memory, and inverter output acting as input to the TSB.*

➤ Here, *memory* is used for ‘pre-loaded constant’, which is part of Trojan trigger logic like PD-HT, DoS-HT, and TB-HT.

➤ In the database, **1st row** defines the unique feature names and **remaining 154 row** defines multiple instances related to various IP designs. Further, the last/final column displays the label (non-infected → 0, and Trojan infected → 1/2/3/4/5/6) of each instance.

➤ These **Trojan infected labels** are defined as:

PD-HT → **1**, DoS-HT → **2**, BE-HT → **3**, DA-HT → **4**, F-HT → **5**, and TB-HT → **6**.

➤ The generated database and a sample of PD-HT infected FIR IP RTL datapath can be found at [10]. Therefore, the developed decision-making RF-classifier based ML model is employed to categorize the HLS based IP designs.

[10] *GitHub Repository*, Available [Online]: https://github.com/NBhui95/HLS_Hardware_Trojans_Detection_using_RF_Classifier.

- The Proposed Methodology (Contd.)

The screenshot shows the GitHub interface for the repository 'NBhui95 / HLS_Hardware_Trojans_Detection_using_RF_Classifier'. The repository is public and has 1 branch and 0 tags. The main branch is selected. The repository contains 6 commits. The commit history table is as follows:

Commit Message	Commit Hash	Time Ago
FIR Datapath IP Design_PD-HT Inserted_VHDL ...	f94c639	8 months ago
FIR RTL Datapath PDHT.pdf		8 months ago
HLS IP Design (Benchmark) Details.pdf		8 months ago
New HLS Trojan Full Database.csv		8 months ago
New Python Script_HLS Trojan Feature Extractio...		8 months ago
README.md		8 months ago

[10] *GitHub Repository*, Available [Online]: https://github.com/NBhui95/HLS_Hardware_Trojans_Detection_using_RF_Classifier.

- The Proposed Methodology (Contd.)

	4*1 Mux	8*1 Mux	16*1 Mux	32*1 Mux	---	Multiplier	Comparator	2*1 Mux	TSB chain	Inverter Ch	Up-counte	Tri-state b	Mux output	Extra_FU_i	Memory	Inverter ou	Label
Covolutional layer-NoI	0	4	2	0	---	1	0	8	0	0	0	0	0	0	0	0	0
Covolutional layer-TI1	0	4	2	0	---	1	1	11	0	20	0	0	1	0	1	0	1
Covolutional layer-TI2	0	4	2	0	---	1	1	11	0	0	0	1	1	0	1	1	2
Covolutional layer-TI3	0	4	2	0	---	1	0	8	0	0	1	0	0	0	0	0	3
Covolutional layer-TI4	0	4	2	0	---	1	1	9	0	0	0	0	0	0	0	0	4
Covolutional layer-TI5	0	4	2	0	---	1	0	9	0	0	0	0	0	1	0	0	5
Covolutional layer-TI6	0	4	2	0	---	1	1	11	48	0	1	0	1	0	1	1	6
FIR-NoI	12	2	2	0	---	2	0	0	0	0	0	0	0	0	0	0	0
FIR-TI1	12	2	2	0	---	2	1	3	0	20	0	0	1	0	1	0	1
FIR-TI2	12	2	2	0	---	2	1	3	0	0	0	1	1	0	1	1	2
FIR-TI3	12	2	2	0	---	2	0	0	0	0	1	0	0	0	0	0	3
FIR-TI4	12	2	2	0	---	2	1	1	0	0	0	0	0	0	0	0	4
FIR-TI5	12	2	2	0	---	2	0	1	0	0	0	0	0	1	0	0	5
FIR-TI6	12	2	2	0	---	2	1	3	48	0	1	0	1	0	1	1	6
MESA-NoI	3	1	0	0	---	4	0	14	0	0	0	0	0	0	0	0	0
MESA-TI1	3	1	0	0	---	4	1	17	0	20	0	0	1	0	1	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---
MESA-TI6	3	1	0	0	---	4	1	17	48	0	1	0	1	0	1	1	6
GLRT-NoI	6	0	0	0	---	1	0	4	0	0	0	0	0	0	0	0	0
GLRT-TI1	6	0	0	0	---	1	1	7	0	20	0	0	1	0	1	0	1
GLRT-TI2	6	0	0	0	---	1	1	7	0	0	0	1	1	0	1	1	2

Fig. 10. Partial view of generated database

[10] *GitHub Repository*, Available [Online]: https://github.com/NBhui95/HLS_Hardware_Trojans_Detection_using_RF_Classifier.

• Result and Analysis

- The proposed HLS hardware Trojan detection framework is implemented on Intel(R) Core(TM) i5-8250U CPU with processor frequency of 1.60GHz and 8GB RAM.
- To implement RF classifier-based ML model in the proposed framework, we have used jupyter notebook within the Anaconda distribution/platform and scikit-learn (free and open-source ML library) with python environment.
- The feature (F_{NoI} and F_{TrI}) extraction process is also performed automatically using python programming language.
The feature extraction time is ~ 59 ms. The respective proposed feature extraction code is publicly available in [10].
- *Note:* The standard hyperparameters utilized in the RF-classifier based ML model are `random_state=42`, `n_estimators=50`, `n_splits=K` (where, $K=5, 6, 7$, and 8 respectively), and `shuffle=True`.

[10] *GitHub Repository*, Available [Online]: https://github.com/NBhui95/HLS_Hardware_Trojans_Detection_using_RF_Classifier.

• Result and Analysis (Contd.)

➤ Here, **Table 3** depicts the results of the proposed Trojan detection approach in terms of standard evaluation parameters with K-fold cross validation (where, K=5,6,7, and 8 respectively).

➤ As evident from the result table, the accuracy of detecting Trojan is 0.90 or 0.92 for specific six state-of-the-art Trojans discussed here.

Note: Here, the number of trees in the forest (i.e., 'n_estimators')=50, which is the part of hyperparameter.

TABLE 3
EVALUATION OF THE PROPOSED DETECTION USING RF
CLASSIFICATION TECHNIQUE IN TERMS OF STANDARD PARAMETERS

K-fold validation	K=5	K=6	K=7	K=8
Accuracy	0.90	0.92	0.90	0.90
Precision	0.92	0.92	0.92	0.92
Recall	0.97	1.00	0.98	0.98
F1-score	0.94	0.96	0.95	0.95
Error rate	0.10	0.08	0.10	0.10

- Result and Analysis (Contd.)

- Further, **Table 4** depicts the results of the proposed Trojan detection approach in terms of standard evaluation parameters with different train-test split ratio (80:20 and 85:15).
- It shows that, the accuracy of detecting Trojan is 0.97 or 0.92 for similar Trojans.

TABLE 4
EVALUATION OF PROPOSED DETECTION USING RF
CLASSIFICATION TECHNIQUE IN TERMS OF STANDARD
PARAMETERS WITH DIFFERENT SPLIT RATIO

Split ratio	80:20	85:15
Accuracy	0.97	0.92
Precision	0.97	0.96
Recall	1.00	0.96
F1-score	0.98	0.96
Error rate	0.03	0.08

• Result and Analysis (Contd.)

TABLE 5

COMPARISON OF PROPOSED RF-BASED HLS HARDWARE TROJAN DETECTION TECHNIQUE WITH DIFFERENT STATE-OF-THE-ART (SOTA) TECHNIQUES (*NOTE: HERE, '×' → NOT DETECTABLE' AND '✓' → DETECTABLE'*)

SOTA hardware Trojan detection techniques (including HLS Trojans)	TL-HLS (DMR based security-aware scheduling) [5]	C to RTL Equivalence checking [11]	TMR based HLS detection [12]	Detection using path delay fingerprint [13]	HLT based detection [14]	Proposed random forest (RF) classifier-based detection technology
PD-HT	×	×	×	×	×	✓
DoS-HT	×	×	×	×	×	✓
BE-HT	×	✓	×	×	✓	✓
DA-HT	×	✓	×	×	×	✓
F-HT	✓	✓	✓	✓	×	✓
TB-HT	×	×	×	×	×	✓

- As shown in **Table 5**, our proposed HLS-Trojan detection technique has the capability to detect all the SOTA HLS Trojans (PD-HT, DoS-HT, BE-HT, DA-HT, F-HT, and TB-HT).
- On the contrary, none of the earlier detection method (SOTA) can identify/detect all of HLS-aided Trojan variances.

• Result and Analysis (Contd.)

- Furthermore, **Table 6** illustrates the HLS Trojan detection status and the detection time related to various HLS IP designs for the proposed RF-classifier based ML Trojan detection method.
- The detection time of the proposed technique is approximately 414 milliseconds.

TABLE 6
TROJAN DETECTION STATUS AND DETECTION TIME

HLS IP designs	Convolutional layer (CNN), FIR, MESA, GLRT, IIR, 4DCT, Sharpening filter (SF), 8-point DCT, LED filter, Blur filter, DWT, Vertical embossment filter (VE), Horizontal embossment filter (HE), Cardiac pacemaker (CP)
HLS Trojan	PD-HT, DoS-HT, BE-HT, DA-HT, F-HT, TB-HT
Detection Status and Time	Yes ~ 414ms

• Conclusion

- A novel HLS hardware Trojan detection mechanism using RF classifier-based ML model is proposed for differentiating the non-infected and HLS Trojan infected IP designs.
- The achieved results are found to obtain high accuracy with $FN = 0\%$ and $FP > 0\%$ respectively, for the six specific class of SOTA HLS Trojans available in the literature.

• References

- [1] M. Rostami, *et. al.*, “A Primer on Hardware Security: Models, Methods, and Metrics,” *Proc. of the IEEE*, vol. 102 (8), pp. 1283-1295, Aug. 2014.
- [2] Y. Jin and Y. Makris, “Hardware trojan detection using path delay fingerprint,” *IEEE HOST*, 2008, pp. 51–57.
- [3] C. Pilato, *et. al.*, “Black-Hat High-Level Synthesis: Myth or Reality?,” *IEEE TVLSI*, vol. 27, no. 4, pp. 913-926, April 2019.
- [4] A. Sengupta, *et. al.*, M-HLS: Malevolent High-Level Synthesis for Watermarked Hardware IPs,” *IEEE Embedded Systems Letters*, vol. 16, no. 4, pp. 497-500, Dec. 2024.
- [5] A. Sengupta, *et. al.*, TL-HLS: Methodology for Low Cost Hardware Trojan Security Aware Scheduling With Optimal Loop Unrolling Factor During High Level Synthesis, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol.36 (4), pp.655-668, 2017.
- [6] A. Sengupta, N. Bhui “Time-Bomb HLS Trojan for Performance Degradation Payload,” *IEEE Design & Test*, vol. 42(5), pp. 83-92, 2025
- [7] M. Rathor *et. al.*, “Revisiting Black-Hat HLS: A Lightweight Countermeasure to HLS-Aided Trojan Attack,” *IEEE Embedded Systems Letters*, Volume: 16, Issue: 2, 2024, pp. 170-173.
- [8] Kento Hasegawa, *et. al.*, “A Hardware-Trojan Classification Method Using Machine Learning at Gate-Level Netlists Based on Trojan Features”, *IEICE Transactions on Fundamentals*, Vol E100-A No.7, 2017.
- [9] R. Yasaei, *et al*, “Hardware Trojan Detection Using Graph Neural Networks,” *IEEE TCAD*, vol. 44, no. 1, pp. 25-38, 2025.
- [10] GitHub Repository, Available [Online]: https://github.com/NBhui95/HLS_Hardware_Trojans_Detection_using_RF_Classifier.
- [11] M. Abderehman, *et. al.*, “BLAST: Belling the Black-Hat High-Level Synthesis Tool,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 11, pp. 3661-3672, 2022.
- [12] A. Sengupta *et. al.*, Exploration of optimal functional Trojan-resistant hardware intellectual property (IP) core designs during high level synthesis, *Microprocessors and Microsystems*, Vol. 103, 2023, 104973.
- [13] A. Sengupta and N. Bhui, “Detecting HLS Hardware Trojans using Random Forest Classifier,” *IEEE Design & Test (D&T)*, vol. 43, no. 3, pp. 37-44, 2026.

Thank You !!!