Early Detection of Hardware Trojans Using Neural Controlled Differential Equations and Analysis of Power Traces

Hasala Senevirathne Rahul Vishwakarma Amin Rezaei

Department of Computer Engineering & Computer Science
California State University, Long Beach

IEEE 18th Dallas Circuits and Systems Conference (DCAS), 2026

The Hardware Trojan Threat

What are Hardware Trojans (HTs)?

Malicious modifications to integrated circuits (ICs)

Composed of a trigger (activation condition) and a payload (malicious effect)

Effects: data leakage, denial of service, system failure

Why is this a growing concern?

Globalized semiconductor supply chain

Reliance on 3rd-party IP cores

Offshore fabrication facilities

Three Circuit States

1. Clean — No Trojan present

2. Dormant — Trojan present, trigger condition not met

3. Active — Trojan triggered, payload executing

Drawback of existing methods

Existing ML methods detect only active Trojans (binary classification).

Dormant Trojans remain undetected.

Motivation: Why Detect Dormant Trojans?

Existing ML-Based Detection Methods

Method	Dormant Detection?
HTM (2021)	**No**
LSTM (2024)	**No**

All prior methods provide only binary classification (Trojan vs. no Trojan) and detect only after activation.

Why dormant detection matters:

Proactive security — detect before activation

Supply chain assurance — identify compromised chips before deployment

Dormant Trojans may still leave subtle side-channel footprints

Our approach:

Three-state classification: Clean, Dormant, Active

Power side-channel analysis (non-invasive)

NCDEs for continuous-time modeling

Background: From Neural ODEs to NCDEs

Neural ODEs (Chen et al., 2018)

Continuous-depth generalization of residual networks

Hidden state evolves as:

$$dh(t)/dt = f\theta(h(t), t)$$

Trajectory entirely determined by $h(t_o)$

✗ **Cannot incorporate new data after t_o**

Neural CDEs (Kidger et al., 2020)

Continuous-time analogue of RNNs

Hidden state driven by input data:

$$h(t) = h(t_o) + \int f\theta(h(s))\, dX(s)$$

fθ: neural network defining dynamics

X(t): continuous control path from data

dX(s): data drives hidden state evolution

✓ **Data continuously influences hidden state**

NCDE Theory: Why It Fits Power Trace Analysis

How NCDEs work:

1. Control path: Discrete observations interpolated via cubic splines → continuous path X(t)

2. Hidden state evolution: CDE converted to ODE form, solved with Runge-Kutta (RK4)

3. Readout: Final hidden state h(T) → prediction via linear layer

Why this suits power traces:

Power consumption is a continuous physical signal

Measurement jitter → irregular sampling

Dormant Trojans → subtle continuous deviations

Advantages over RNNs/LSTMs:

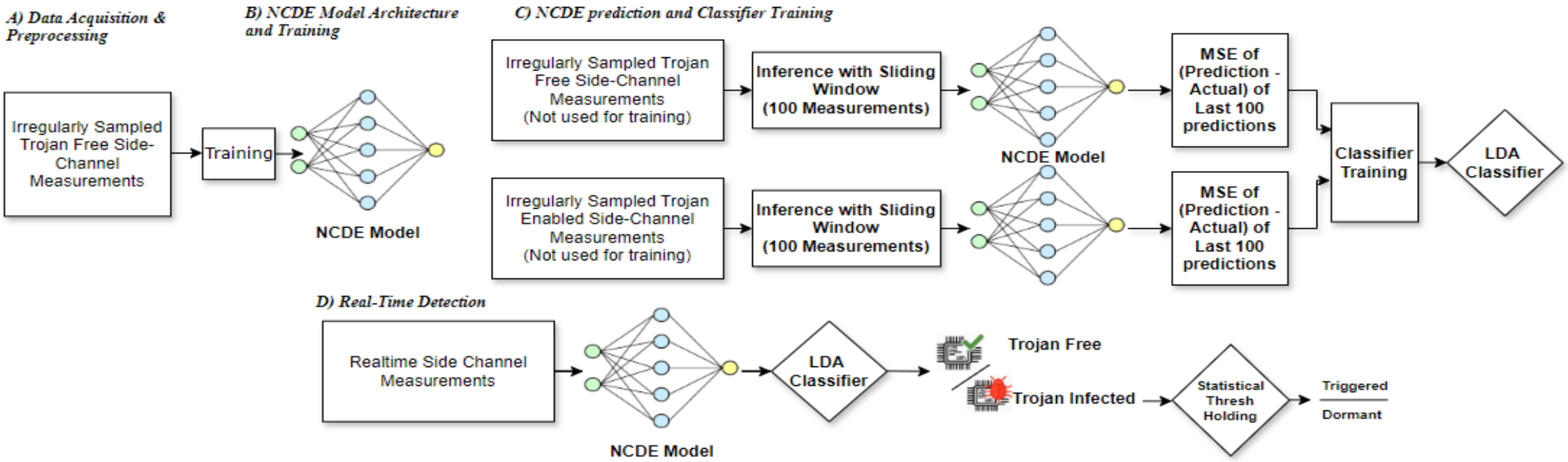
Irregular sampling: Handled natively via continuous control path — no imputation needed

Inter-sample dynamics: Captures behavior between discrete sample points

Memory efficiency: O(L+H) vs. O(L×H) for RNNs

RNNs/LSTMs process data at discrete steps and need workarounds for irregular data. NCDEs handle this natively via the continuous control path.

HOODOO Framework Overview



A: Preprocessing

Normalize, map time to [0,1], cubic spline interpolation

B: NCDE Training

Train on Trojan-free data only; learn nominal behavior

C: Classifier

Compute MSE on labeled data; train LDA classifier

D: Detection

Real-time three-state classification

Three-State Classification

Core Idea:

NCDE trained on clean data only learns nominal power behavior

On Trojan-infected traces, prediction errors increase

Sliding window (W=50): predict next value, compute squared error

Aggregate into Mean Squared Error (MSE)

LDA Classifier:

Single feature: MSE value

Calibrated on labeled clean + infected traces

Determines optimal boundary b_LDA

Classification Logic

No Trojan if $MSE \leq b\_LDA$

Dormant if $b\_LDA < MSE \leq T\_trig$

Active if $MSE > T\_trig$

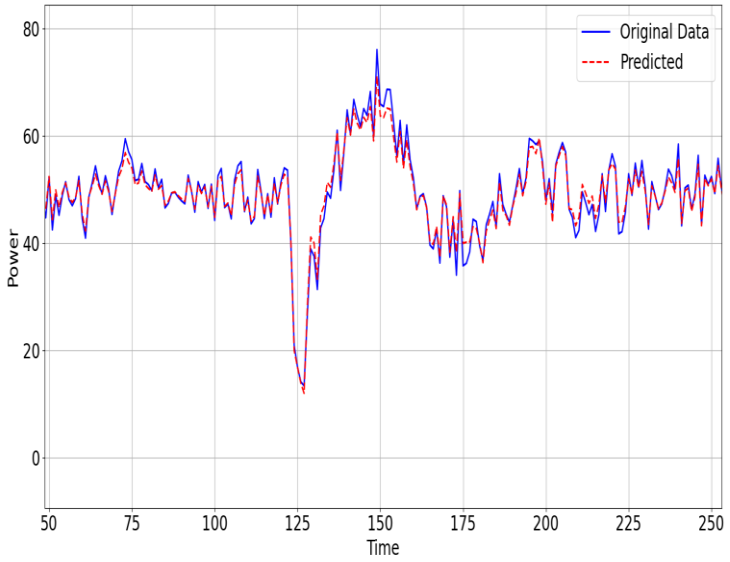
What makes this different

First three-state HT classification

Learns “normal,” flags deviations

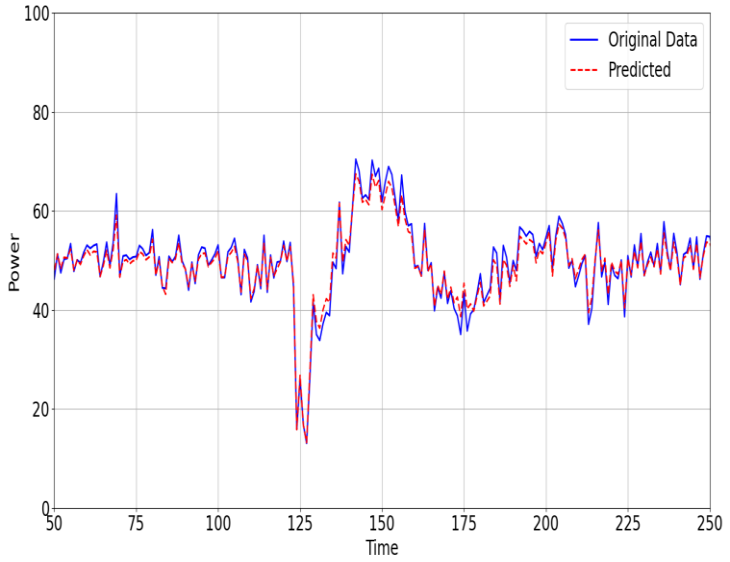
Dormant vs. active by deviation magnitude

NCDE Predictions Under Different HT Conditions



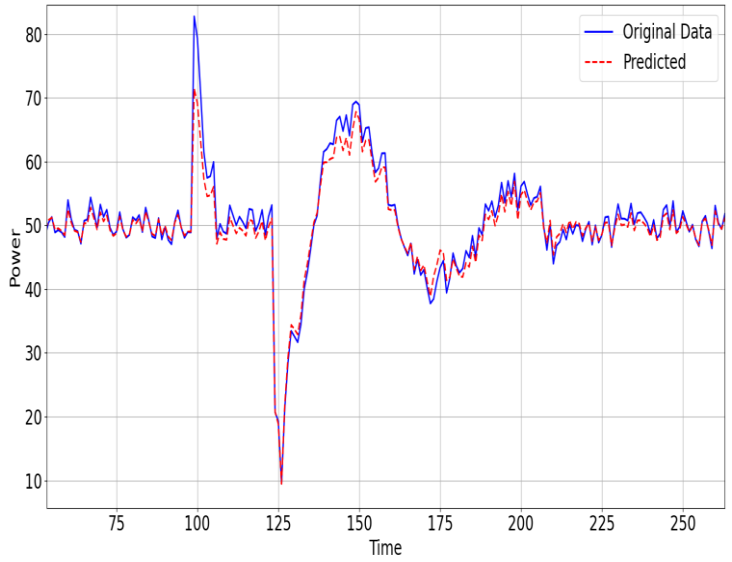
(a) No Trojan

Tight overlap — model accurately captures nominal behavior



(b) Trojan Dormant

Subtle deviations emerge between predicted and actual values

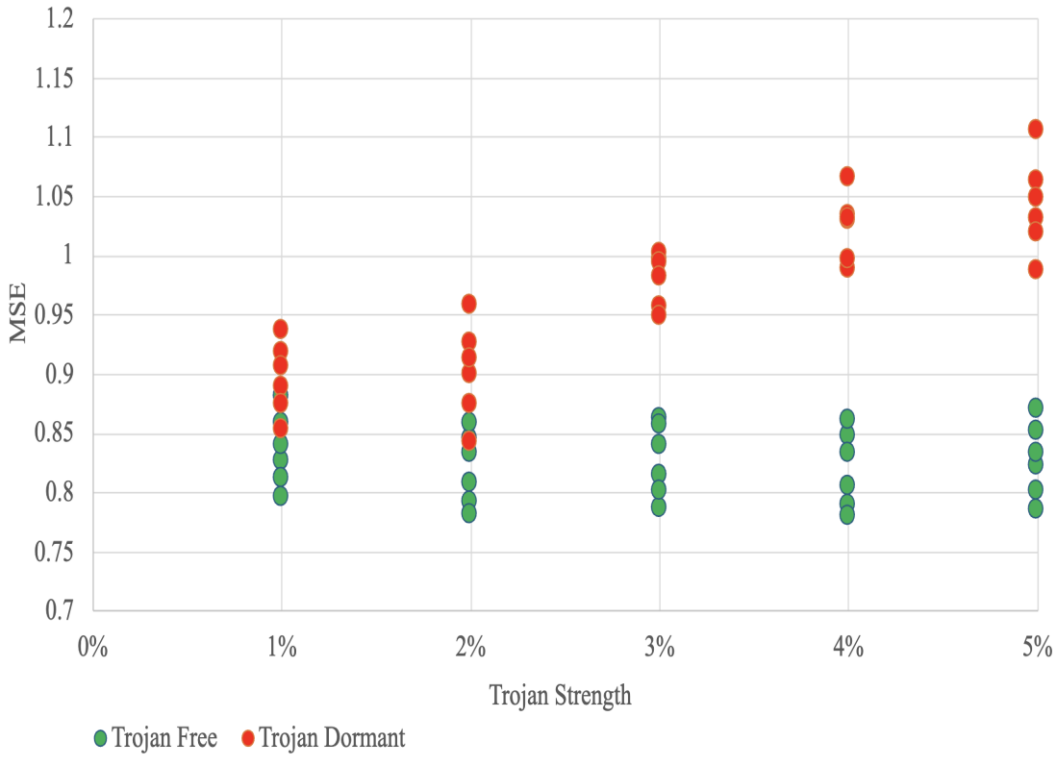


(c) Trojan Triggered

Pronounced deviations at trigger points — clearly detectable

Blue: Original Data **Red dashed: NCDE Predicted** — Higher deviation ⇒ higher MSE ⇒ Trojan indication

Results and Comparison



Detection Sensitivity

MSE vs. Trojan Strength (1–5%)

Green: Trojan-free **Red**: Dormant

At ≥3%, clusters separate reliably.

Method	3-State?	Dormant	Active
MLNN	No	—	85.0%
LSTM	No	—	86.8%
HTM	No	—	92.2%
		—	
Ours (1%)	**Yes**	55.7%	92.4%
Ours (2%)	**Yes**	62.5%	94.6%
Ours (3%)	**Yes**	80.2%	95.4%
Ours (4%)	**Yes**	88.3%	99.3%
Ours (5%)	**Yes**	92.2%	100%

Observations:

Active detection: competitive at ≥3%

Dormant detection: not available in prior work

Note: methods use different input modalities

Limitations, Future Work, and Conclusion

Limitations:

Below 3% peak amplitude → detection degrades

Gaussian noise as proxy; real dormant Trojans may have non-Gaussian signatures

MSE as sole feature; NCDE latent states could be richer

T_triggered not calibrated on real data

Future Directions:

Multi-modal side-channel data (power + EM + timing)

Richer anomaly features from NCDE hidden states

Cross-chip generalization and process variation

Transfer learning for new hardware designs

Conclusion

NCDE-based three-state classification offers dormant Trojan detection — absent in prior methods — with competitive active detection at ≥3% perturbation levels.

Acknowledgment

NSF Award No. 2245247

Contact

hasala.senevirathne01@student.csulb.edu