IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 27 February 2024, accepted 21 March 2024, date of publication 1 April 2024, date of current version 10 April 2024.

*Digital Object Identifier 10.1109/ACCESS.2024.3383834*

RESEARCH ARTICLE

# PRISTINE: An Emulation Platform for PCB-Level Hardware Trojans

**JUNJUN HUAN[1], SHUBHRA DEB PAUL[1], (Member, IEEE), SOUMYAJIT MANDAL[2], (Senior Member, IEEE), AND SWARUP BHUNIA[1], (Fellow, IEEE)**

[1]Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL 32611, USA
[2]Instrumentation Division, Brookhaven National Laboratory, Upton, NY 11973, USA

Corresponding author: Junjun Huan (junjun.huan@ufl.edu)

**ABSTRACT** Printed circuit Boards (PCBs) are becoming increasingly vulnerable to malicious design alteration, also known as Trojan attacks, due to a distributed business model that often involves various untrusted parties. Such attacks can be mounted at various stages in the PCB life cycle. The relative ease of alteration of PCB hardware even after fabrication (due to physical access to surface-mounted critical components and traces) makes them attractive for an adversary to manipulate their functional/physical behavior for malicious intent. There is a growing need to explore viable Trojan attacks in a PCB, analyze their functional and physical characteristics (e.g., impact on power or delay), and study the effectiveness of countermeasures against these attacks. While simulation-based approaches for PCB Trojan insertion are effective at creating a large population of possible Trojans, they fail to provide functional feasibility analysis with a realistic workload for a trigger circuit. Also, they cannot estimate a Trojan's side-channel footprint due to the unavailability of physical models of diverse PCB components. To address these deficiencies, in this paper, we present PRISTINE, a PCB-level emulation system for any integrity or physical tampering issues, specifically, hardware Trojan insertion. The need for building such an emulation platform to resolve PCB trust issues in the supply chain is also surveyed and discussed. Both custom Hardware Hacking (HaHa) boards and multiple commercial PCBs are then used to test the ability of the proposed system to emulate various hardware Trojans specially designed to exploit board-specific hardware characteristics. Experimental results on emulated board-level Trojans show that a wide range of Trojans can be successfully activated, thus enabling the expected payload effects on both types of boards to be studied and quantified. The resulting data are further analyzed to create PCB-level Trojan benchmarks. In particular, a comparative evaluation of the experimental results is used to propose a risk level metric that quantifies the probability of detection and degree of payload impact of each Trojan on a given commercial PCB.

**INDEX TERMS** Hardware Trojans, PCB security, Trojan emulation, hardware tampering, supply chain security.

## I. INTRODUCTION

The ubiquity of PCB Trojan threats justifies the need to i) create benchmarks for PCB-level hardware Trojans, and ii) explore possible countermeasures and their efficacy in detecting and avoiding them. While extensive research has been conducted on IC-level Trojans [1], [2], [3], threat models of PCB-level Trojans, which are significantly different, have not been sufficiently studied. One major difference between the two types is that PCB Trojans can be inserted *after* board fabrication and assembly [4], [5], [6]. Examples of such in-field Trojan attacks include Xbox hacking by using a modchip [7], DRM-key bypassing by tampering of a PCB within a television set [8], and JTAG data bus hijacking on a PCB [9]. Another difference is the wide variety of substrates, fabrication methods, and components used by PCBs, which limits the applicability of inspection-based Trojan-detection methods.

The associate editor coordinating the review of this manuscript and approving it for publication was Jiafeng Xie.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

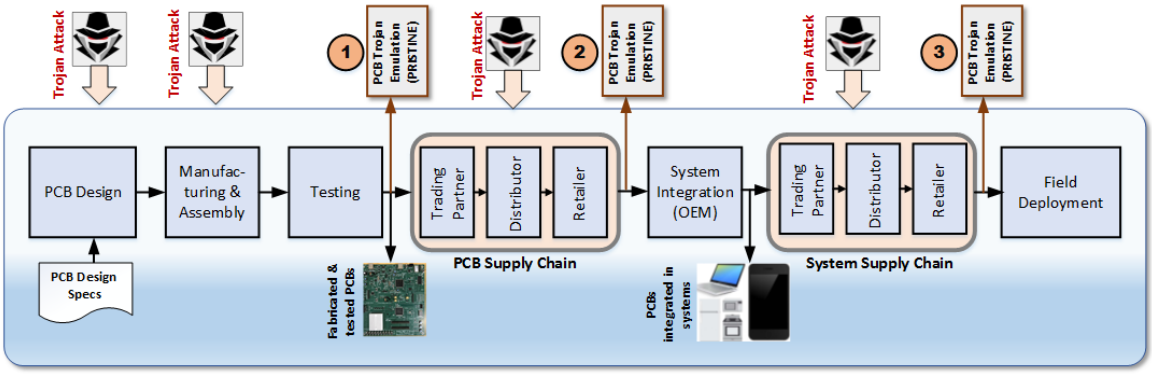


**FIGURE 1.** Modern PCB supply chain eco-system and the stages where Trojan attacks can be mounted. The usage of a PCB-level Trojan emulator can be used at various stages, as shown.

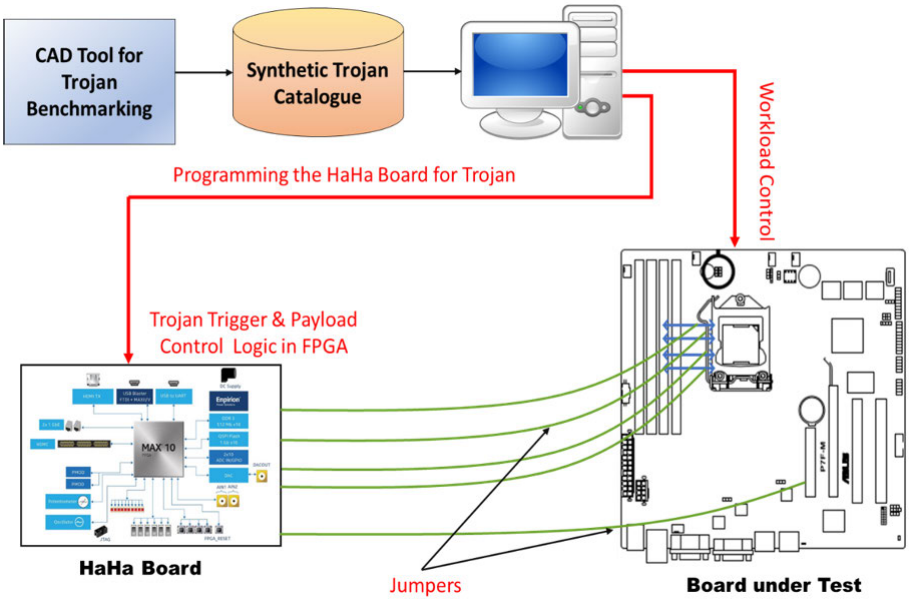


**FIGURE 2.** Board-level Trojan prototyping and evaluation, consisting of a custom-designed FPGA board (named HaHa) that implements diverse Trojan trigger/payload conditions and a computer that controls the workload in the Board under Test (BUT).

Another practical challenge is the lack of accessible design resources (schematic and layout) for many commercial PCBs, such as computer motherboards. The design documents for such products are generally not publicly available, which makes it challenging for security researchers to completely understand the entire design at the system level to identify potential vulnerabilities. Therefore, in this paper, we present PRISTINE, a **Pri**nted Circuit Board level **T**rojan **In**sertion **E**mulation **S**ystem that can provide benchmarks for PCB-level hardware Trojans using both a custom PCB (known as HaHa) designed for security research [10], [11] and commercial PCBs such as motherboards and single-board computers. Figure 2 provides an overview of PRISTINE – the proposed PCB-level Trojan Insertion Emulation platform. PRISTINE uses the custom-designed HaHa board to implement diverse Trojan trigger/payload conditions on the BUT, which is also connected to a computer that provides various workloads. The BUT can either be another HaHa board or a commercial PCB. In addition, an attached oscilloscope (not shown) can be used to measure currents on the BUT using a sense resistor on the HaHa board. Each Trojan design was customized for both types of BUT. The Trojans were then tested on our emulation platform and evaluated using a risk metric that combines probability of detection with degree of severity of payload outcomes. The test results and risk metrics can be used for: i) efficiently rank-ordering PCB Trojan threats, and ii) devising countermeasures for the most relevant threats.

The rest of the paper is organized as follows. Section II presents the theoretical background of hardware Trojan

attacks on PCBs. Section III describes existing hardware emulation platforms in the market along with their deficiencies in emulating PCB-level hardware Trojans. The need for the presented emulation platform is also explained. Section IV explains the methodology of the proposed PCB Trojan emulation platform. Section V presents the experimental results, while Section VI discusses possible extensions of the emulation platform to address limitations encountered in the experiments (e.g., by allowing insertion of a wider variety of Trojans). Section VII summarizes the paper and discusses our plan for future work.

## II. BACKGROUND AND MOTIVATION

### A. PCB-LEVEL HARDWARE TROJANS

At the PCB level, hardware Trojan insertion typically starts by exploiting the vulnerabilities of either electronic components on the board or peripheral devices externally connected to the board's input/output (I/O) modules. Vulnerable nodes are then maliciously modified to force functional alterations and information leakage from the chosen components and devices [8]. Effective PCB-level Trojans should be rarely triggered and detected to escape from functional or parametric tests performed during PCB inspection and testing.

PCB-level Trojans can be activated both through internal triggers such as IC operating states or user actions (e.g., toggling switches or pressing push-buttons on the board) and external triggers such as control signals received either over a wireless link or directly from peripheral devices. Multiple triggers can be bundled through certain circuits to increase the rarity of Trojan activation. One example is using an AND gate to trigger a Trojan. Each Trojan will only be activated if all trigger conditions are satisfied. Both internal and external triggers are implemented in our experiments to test the flexibility of our Trojan emulator. Trojan triggers at the PCB level can also be implemented based on components' electrical parameters instead of their functions. A typical example is combining a comparator circuit with a capacitor to trigger a Trojan whenever the average voltage value at a trigger point rises above a preset threshold [12].

The payload of any Trojan constitutes its core functionality. As introduced in [8], PCB Trojan payloads can be classified into three broad categories. The first type of payload maliciously modifies the functions of on-board ICs, other components on the PCB, or peripheral devices. This may lead to further impacts like shutdown of individual components or malfunction of the entire PCB system. The second type of payload reconfigures the PCB to leak secret information to unauthorized parties. Finally, the third type of payload is implemented through a side channel attack. Side channels contain information that is not intended to be communicated between parties. Most commonly used side-channel data is collected through analysis of power, timing, or electromagnetic emission features of the on-board ICs, other electronic components, and peripheral devices. Fig. 3 demonstrates how a simple bit-flipping Trojan payload

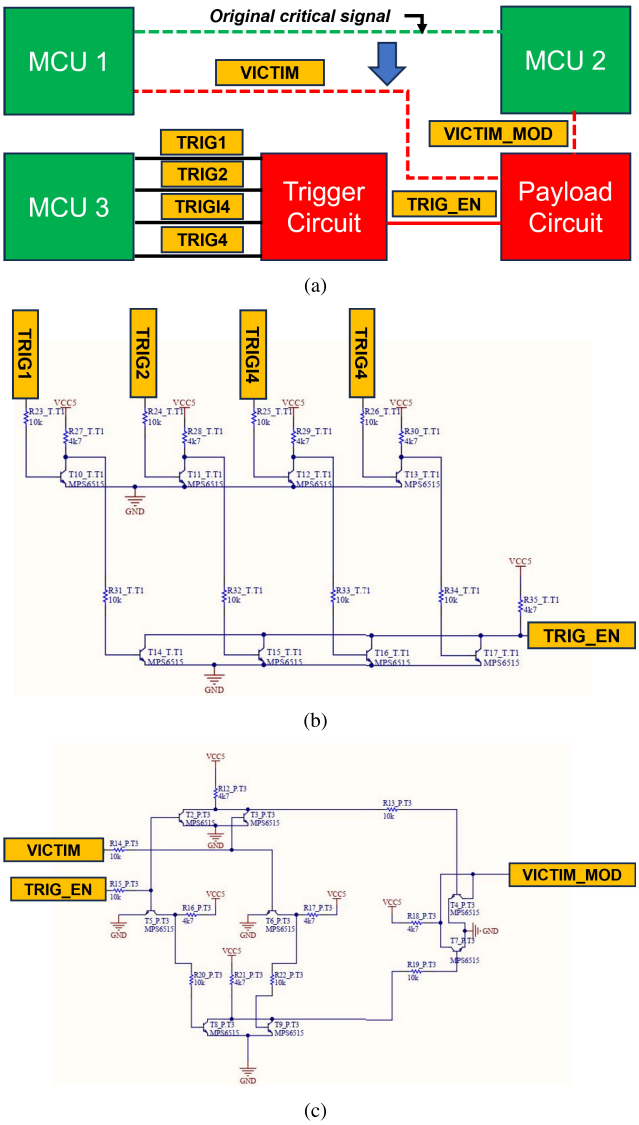


**FIGURE 3.** Example of a PCB-level Trojan implanted with simple transistors and resistors. (a) Trojan system architecture, (b) trigger circuit, and (c) payload circuit.

(an example of the first payload category) can be activated through a 3-input AND-gate Trojan trigger. The impact of this particular Trojan is to maliciously modify a given on-board signal (labeled *Victim*) into its inverted version, *Victim_Mod*.

### B. THE PCB LIFE-CYCLE AND TROJAN THREAT MODELS

Fig. 4 summarizes the typical life cycle of a PCB in the form of a flowchart. The figure shows that several phases of this process are vulnerable to Trojan insertion [8]. The life cycle begins with PCB design, fabrication, and system integration phases and then transitions to normal operation (the "in service" phase). The design phase includes several sub-phases (including PCB specification development, circuit, schematic, and layout design, and verification/validation), while the fabrication phase includes

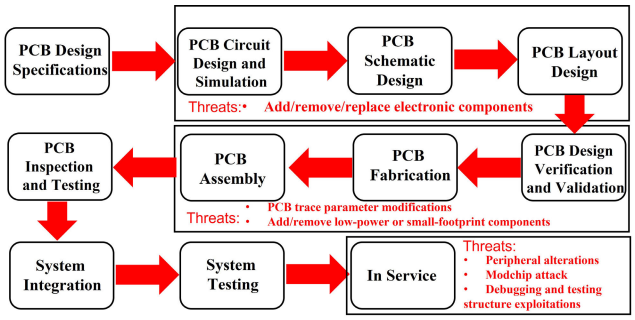


**FIGURE 4.** Flowchart of a typical PCB life cycle, including potential Trojan insertion phases.

PCB fabrication, assembly, inspection, and testing. The next phase includes system integration and system testing. Both the design specification and system integration phases are not subject to attacks from hardware Trojans, and thus can be assumed to be unaffected as long as the design house is trustworthy [8]. On the other hand, most PCB fabrication and assembly work is outsourced to external foundries, which makes both the fabrication and assembly processes potential sources of PCB Trojan attacks. Besides, unlike for ICs, the post-fabrication phase of a PCB also involves Trojan threat sources, including peripheral device alterations, modchip attacks, and exploitation of built-in debugging and testing infrastructure (e.g., JTAG ports). Thus, if we assume that the design house is trustworthy, hardware Trojans can be inserted during all phases of the PCB life cycle except for the design specification and system integration phases. For example, attackers in PCB fabrication houses can exploit physical design vulnerabilities, including electrical parameters of the traces and data buses of on-board ICs (such as their resistance, capacitance, inductance, characteristic impedance, or signaling levels) as well as inputs/outputs of peripheral devices, and then insert hardware Trojans to disrupt the functions of specific electronic components and devices and/or to stealthily leak information.

If the design house is also not trustworthy, PCB Trojan design becomes even more flexible and need not be restricted to simple trace or IC modifications. For example, rogue PCB designers can insert complex Trojans into the board design by including additional passive or active components. An example of this type of PCB Trojan, namely a modified Arduino fan-speed controller that prevents accurate temperature sensing via malicious insertion of a simple circuit (resistor, capacitor, and PMOS transistor), was presented in [13].

### C. TAXONOMY OF PCB-LEVEL TROJANS

Identification and analysis of a taxonomy for PCB-level hardware Trojans is necessary to help create PCB Trojan benchmarks and discover countermeasures. To the best of our knowledge, a taxonomy for PCB Trojans was first proposed in [12]. Fig. 5 shows a modified taxonomy that is also valid for commercial PCBs. The modules implemented in our hardware Trojan emulation system are labelled through dotted boxes. As shown in the figure, PCB Trojans can be classified using several criteria (insertion phase, abstraction level, activation mechanism, payload, location, and type of modification). These classes are similar to those used in commonly-used taxonomies for IC Trojans [14]. However, the proposed taxonomy includes additional categories (e.g., a "post-fabrication" insertion phase) to account for PCB-level Trojans that are inserted after completion of the fabrication and assembly processes; this is one of the most important Trojan insertion pathways for commercial boards.

Post-fabrication PCB modifications can be used to implement a variety of Trojan triggers. An example of experimentally implementing a post-fabrication Trojan was described in [15]. In this work, an Xbox console was hacked by tampering with its copyright restrictions such that games could be played illegally. Different PCB layers can be used as locations for Trojan insertion. The top and bottom layers can serve as target locations for PCB insertion if a layout reference is not easily available. However, when a layout reference is accessible, PCB Trojans are mostly inserted in the inner layers of the board to evade visual inspections and testing. Post-fabrication PCB modifications can also be used as Trojan triggers. An example of this type of modification was demonstrated on a microcontroller development board (Arduino Uno) [13]. In this case, passive components were added and traces were rerouted over small areas of the PCB inner layers to corrupt the functionality of the board.

### D. RELATED WORK ON PCB TROJAN BENCHMARKING

In this section, we consider the problem of benchmarking the performance of a PCB that may include one or more Trojans in the form of netlists, specifications, schematics, or layouts. The authors of [12] and [16] proposed a software CAD-tool flow that is capable of scanning and simulating rarely-switched nodes on an uninfected sample PCB and then inserting hardware Trojans into the board in the form of netlists that are already registered in a Trojan database. The modified PCB then behaves as an infected PCB in real-life situations. Importantly, the modified design can be simulated using a board-level software simulator, thus allowing us to benchmark the inserted Trojan by observing its effects. Thus, the proposed procedure enables an unbiased evaluation of the relative merits of various countermeasures for PCB trust issues. A simulation-only implementation is, however, not adequate for the following four reasons. Firstly, many commercial PCB manufacturers usually label PCB design documents as classified resources and do not release them to the public. As a result, reverse engineering is the only remaining option for recovering the complete PCB netlist. Secondly, no free or low-cost CAD software tools offer board-level post-layout simulation functionality. To the best of our knowledge, the only PCB design software with a built-in post-layout simulator is Cadence

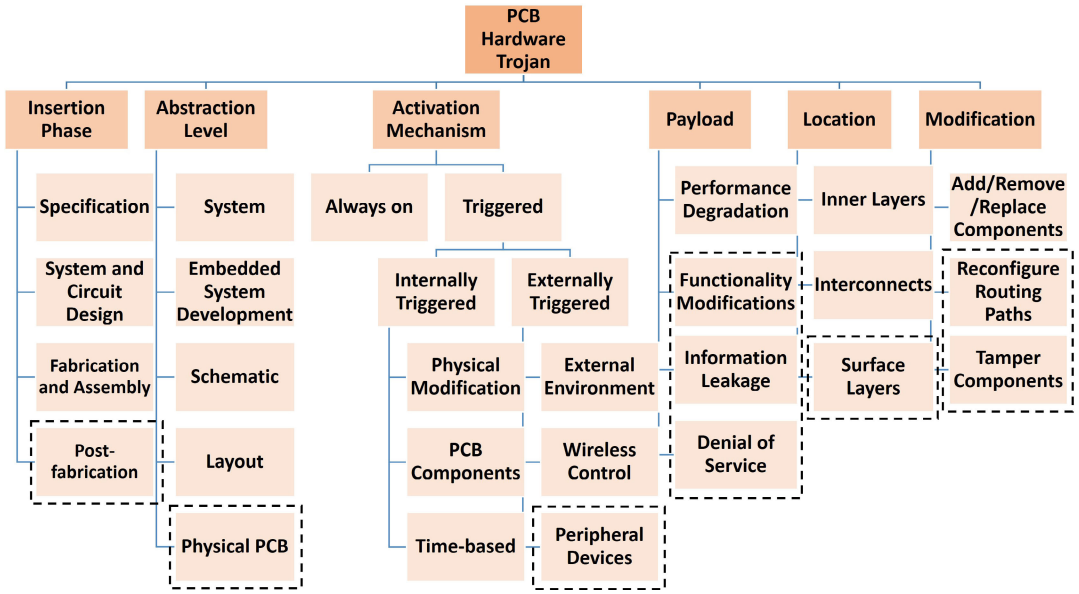


**FIGURE 5.** Taxonomy of PCB hardware Trojans.

Allegro [17]. Thirdly, some of the major IC vendors (e.g., Texas Instruments) do not provide simulation models for standard SPICE-based simulation tools but instead use their own custom circuit simulators [18], which makes even schematic-based simulations challenging. Finally, deriving the correct simulation test bench for a PCB netlist is generally a challenging task that has proven resistant to automation. Thus, in this paper we build an experimental PCB Trojan emulation platform for both custom and commercial PCBs. The proposed platform addresses a variety of trust issues in the PCB supply chain by enabling controlled insertion and benchmarking of user-defined Trojans.

## III. NEED FOR HARDWARE EMULATION FOR PCB-LEVEL TROJAN ATTACKS

In this section we describe existing hardware emulation platforms, their deficiencies in emulating hardware Trojans at the PCB level, and the need for the proposed emulation platform.

### A. HARDWARE EMULATION PLATFORMS

As demonstrated in Table 1, hardware emulators can be classified into two main categories, namely commercial-based general purpose emulators and research-based hardware Trojan emulators. General purpose emulators are commercial products that have been upgraded for decades to improve scalability, reliability, and speed of hardware/software verification, while hardware Trojan emulators are still being explored in research specifically for modeling Hardware Trojan attacks and discovering corresponding detection techniques.

The general-purpose emulation platform market primarily consists of three main competitors, namely Cadence's Palladium (Palladium Z1 and Z2), Synopsys' Zebu (Zebu EP1, Empower and Server 4) and Siemens' Veloce (Strato). Three main use cases of each emulation platform are listed along with design/Trojan size in Table 1 [22], [23], [24]. All three emulators exclusively exhibit functions of hardware/software and SoC verification and debug as well as in-circuit emulation when connected to external hardware. Apart from these two main features, Palladium and Zebu also offer the feature of early software development and bring-up and Veloce specially provides transaction-based acceleration. In terms of design capacity, all three emulation platforms support designs with over a billion gates, with Palladium and Veloce scaling to approximately 15 billion gates.

Several research-based emulation platforms for modeling hardware Trojan attacks and identifying detection methods are presented in recent publications [19], [20], [21] and listed in Table 1. Wu et al. presented an emulation system to detect and prevent hardware Trojan attacks on FPGA/ASIC chips [19]. Four different real-life hardware Trojans were modelled and inserted in the system for Trojan detection verification, including modification of RTL and layout design, IC modification, malicious reprogramming of systems in an FPGA, and leakage of information. Bolchini et al. described their work on microprocessor-based hardware Trojan attacks and a detection emulation system [20]. The group implemented activation analysis of hardware Trojans on a running software and evaluated potential detection techniques against three types of Trojans: functionality modification, denial-of-service and information leakage. Pearce et al. [21] presented 3 different types of hardware Trojan examples to be inserted into their PCB-level PLC-based test bed to evaluate the detection method of using side-channel loopbacks.

**TABLE 1.** Primary features of different hardware emulation platforms.

| Emulator Platform | | Key Features | |
|---|---|---|---|
| | | Use Cases | Design/Trojan Size |
| General Purpose Emulator | Palladium | Early software development<br>Hardware/software and SoC verification and debug<br>In-circuit emulation | Up to 18.4B gates |
| | Zebu | Software bring-up<br>Hardware/software and SoC debug<br>In-circuit emulation | Up to 2B gates |
| | Veloce | SoC emulation<br>Transaction-based acceleration<br>In-circuit emulation | Up to 15B gates |
| Hardware Trojan Attack or Detection Emulator. | Wu et al. [19] | Detection of hardware Trojans during testing or system operation of an ASIC chip<br>Detection of hardware Trojans during testing or system operation of an FPGA chip | 4 types of Trojans |
| | Bolchini et al. [20] | Analysis of activating a given set of hardware Trojan Horses (HWTs) by a running software on a microprocessor<br>Assessment of detection techniques against the HWTs in microprocessors | 3 types of Trojans |
| | Pearce et al. [21] | Detecting hardware Trojans in PLC-based PCBs using side-channel loopbacks | 3 types of Trojans |
| | PRISTINE | Activation and impact analysis of hardware Trojans on a custom PCB<br>Activation and impact analysis of hardware Trojans on a commercial PCB such as a motherboard or a single board computer | 7 types of Trojans |

### B. DEFICIENCIES OF EXISTING PLATFORMS

The existing emulation platforms described in the previous section have several deficiencies in precisely modelling PCB Trojans and their effects on the PCB. Firstly, the computer architecture, central processing unit (CPU), peripheral chips, and other components of a commercial board are hard to model using a general-purpose emulator, because the circuitry or firmware design of almost every computer architecture or peripheral IC chip is produced under a proprietary process and is not easily accessible without being purchased from the design company. In addition, when modelled through a component of the general computer architecture, a CPU or other chips may not precisely demonstrate the effect of a specific Trojan trigger or payload. For example, most motherboards have an IC to handle low-speed input/output devices for serial and parallel port control, keyboard and mouse control, voltage limiting, real-time clock control, and temperature and CPU fan control. This chip can be tapered to trigger malfunction of the motherboard. However, the IO chips of some older motherboards may have only the basic port control functionality and cannot be modelled through a general design. Secondly, existing general-purpose emulators may not be interfaced with peripheral devices like USB keyboards or mice, audio speakers, CPU fans, or Ethernet networks that usually serve as sources of Trojan triggers or payloads. Thirdly, even though most digital PCB Trojans can be easily generated through RTL-level design in the existing general-purpose emulators, analog Trojans like the Trojan payload in Fig. 9(f), which requires an analog operational amplifier IC, are not accessible and cannot be added to the PCB Trojan benchmark. Fourthly, it is much more cost-effective to implement the PRISTINE emulator due to its low production costs and use of open-source hardware. Fifthly, most other existing research-based hardware Trojan emulators [19], [20] are specifically designed for either FPGA or microprocessor ICs, while PRISTINE allows for modelling system-level or circuit-level Trojan attacks on a PCB during the post-fabrication phase of the PCB life cycle. Finally, even though some research groups [15], [21], [25], [26] presented designs of PCB-level Trojan attacks to evaluate their Trojan detection methods, they did not focus on developing a hardware Trojan emulator with enough Trojan examples and only used custom PCBs as hardware Trojan targets. By contrast, PRISTINE models 7 types of hardware Trojans on both custom and commercial boards (such as motherboards and single-board computers), thus enabling more general applications.

### C. MOTIVATION FOR PRISTINE

In this section, we summarize our motivations for designing the proposed PCB Trojan emulation platform (PRISTINE):

1) Explore the threat of board-level malicious modifications. Malicious alterations of system or board-level electronics have not been studied as much as those of chip- or transistor-level electronic components. Even though PCB-level Trojan attacks may not be as challenging to detect and overcome as IC-level Trojans, to figure out potential solutions still requires effort to emulate various hardware Trojan attacks, analyze the probability of detection, and estimate the possible impact of each attack.
2) Collect PCB-based hardware Trojan attack data for PCB Trojan benchmarking. The PCB Trojan attacks are modelled to emulate real-life hardware Trojan attacks (including analog ones) in a system that can be easily interfaced with peripheral devices. The resulting data can be processed and analyzed to help generate countermeasures to various attacks and solve the lack

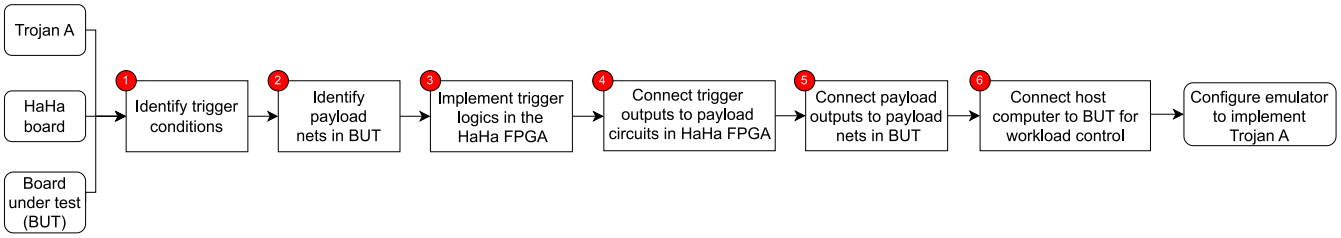


**FIGURE 6.** Summary of the process used by the PRISTINE PCB Trojan emulator for emulating hardware Trojan insertion.

of trustworthiness problem in the PCB life cycle that primarily arises during the post-fabrication phase.

3) Provide a cost-effective platform for PCB Trojan emulation. Our system provides a low-cost FPGA-based platform to generate Trojan triggers and payloads for PCB-level Trojan emulation. The device costs 10-100 times less than typical hardware emulation platforms on the market. Also, only easy-to-use jumper wires are needed to interface the emulator with the target victim board for Trojan insertion.

## IV. PRISTINE PLATFORM ARCHITECTURE AND EMULATION METHODOLOGY

In this section, we present the proposed Trojan insertion emulation process of our PCB Trojan emulator, the emulator architecture, and the Trojan models that it supports.

Fig. 6 summarizes the process used by the PRISTINE PCB Trojan emulator for emulating the insertion of a Trojan, *A*, into a BUT using the HaHa board as the Trojan generator. The Trojan trigger conditions can be identified by analyzing the mechanisms of the input peripheral devices connected to the BUT. Locations of vulnerabilities on the BUT are then investigated to identify the board nets where Trojan *A* should be inserted. Trigger logic is then executed in the HaHa board under the trigger conditions to activate the payload circuits by connecting the trigger outputs to the payload circuits. The payload outputs are then delivered to the target nets on the BUT through a wired connection. The workload of the BUT is controlled by the host computer. Finally, the emulator is configured to implement Trojan *A* on the payload nets of the BUT.

### A. SYSTEM ARCHITECTURE

The architecture of our system for Trojan insertion emulation experiments, which can utilize either a custom HaHa board, a computer motherboard, or a single-board computer as the BUT, is depicted in Fig. 7. The system inserts Trojans into the BUT (or victim board) by using a custom HaHa board as a Trojan generator. The latter uses an on-board FPGA to implement the payload circuitry for a functional Trojan. The trigger input sources of each Trojan are derived either from ICs on the victim board or from peripheral devices. The output of the trigger circuit transmits a "Trigger Enable"

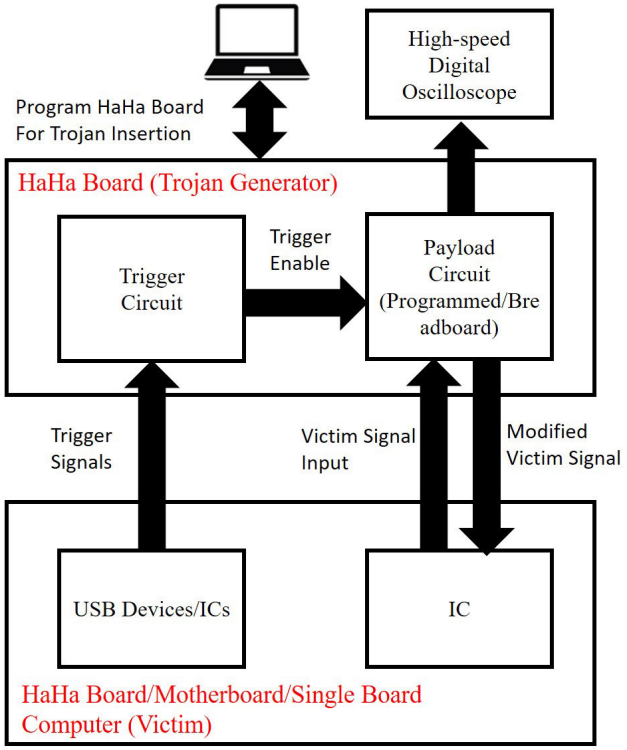


**FIGURE 7.** High-level view of the system architecture used for Trojan insertion experiments.

signal that can activate each Trojan via its own FPGA-based payload circuit; the latter can utilize input signals from other electronic components and/or devices of the victim PCB. Through the Trojan payload circuit, both communication or side channel information leakage and Trojans of the malicious memory/data write type can be implemented on the victim nodes. Their impacts can include automatic overriding of the state of on-board LEDs, corruption of the readings of sensor ICs or peripheral device parameters with false values, freezing or shutdown of the entire system, or unauthorized readout of secret data bus information from the victim PCB, peripheral device, and/or other electronic components. A suitable data acquisition (DAQ) system, such as a high-speed digital oscilloscope, can be used to probe the modified and/or leakage signals of the victim component or device.

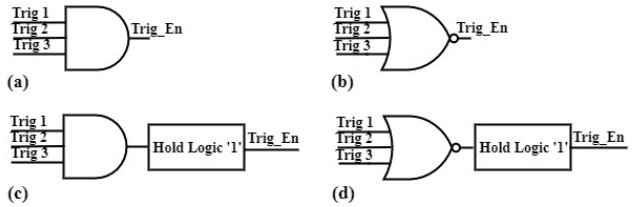


**FIGURE 8.** Triggers used for emulating PCB-level Trojan insertion using the custom HaHa board: (a) an AND gate, and (b) a NOR gate. Triggers used for emulating PCB-level Trojan insertion using commercial computer motherboards: (c) an AND gate plus a circuit to hold logic '1', and (d) a NOR gate plus a circuit to hold logic '1'.

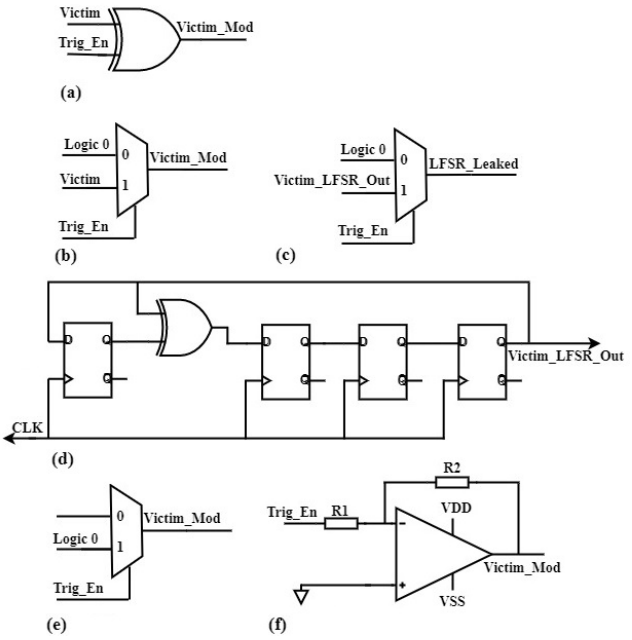


**FIGURE 9.** Trojan payloads used for emulation with the custom HaHa board: (a) an XOR gate, (b) a multiplexer that outputs the victim signal when enabled, and (c) a multiplexer with a linear feedback shift register (LSFR) circuit at the inputs that outputs the victim signal when enabled. (d) The LFSR circuit used in (c). Trojan payloads used for emulation with commercial computer motherboards: (e) a multiplexer that outputs logic '0' when enabled, (f) an operational amplifier (op-amp) circuit used to generate an analog DC signal to modify the input of an on-board ADC.

### B. PCB-LEVEL TROJAN MODELS, DESIGNS, AND SCHEMATICS

#### 1) PCB-LEVEL TROJAN MODELS ON THE CUSTOM HAHA BOARD

Fig. 8 and Fig. 9 list two different types of Trojan trigger circuits (Figs. 8(a)-(b)) and three different Trojan payload circuits (Figs. 9(a)-(d)) that were used to generate hardware Trojans using the custom HaHa board. The trigger circuits consist of an AND gate or a NOR gate. The AND gate trigger is enabled only when all its inputs become logic high. By contrast, the NOR gate trigger is enabled only when all inputs are logic low. Both circuits were used to trigger the activation of the Trojan payloads shown in Fig. 9. The number of trigger inputs determines the probability of the Trojan being activated. These inputs (denoted by Trig_1, Trig_2 and Trig_3) are derived from a second HaHa board that serves as the victim, as shown in the system architecture diagram (Fig. 7).

Three different payload circuits, as shown in Figs. 9(a)-(c), were designed to perform the functions of three types of Trojans. Each Trojan is only activated when its trigger conditions are satisfied. The circuit in Fig. 9(a) uses an XOR gate to invert each input data bit, thus implementing a Trojan that operates as a malicious memory/data write element. Another type of Trojan uses a multiplexer as the payload circuit (as shown in Fig. 9(b)) to leak either communication or side-channel data. This type of Trojan can be made less detectable by randomizing the input information with a linear feedback shift register (LFSR) circuit, as shown in Figs. 9(c)-(d). Different combinations of these trigger and payload circuits can be used to design more sophisticated Trojans that maliciously modify data bus memory and/or automatically leak data bus information on the victim board based on the trigger inputs.

#### 2) PCB-LEVEL TROJAN MODELS ON COMMERCIAL DESKTOP MOTHERBOARDS

Trigger circuits for commercial desktop motherboards were designed by adding an extra circuit that holds the trigger output value as long as the trigger logic generates a logic high. Two such Trojan trigger circuits are depicted in Figs. 8(c)-(d). The extra circuit acts as a state machine that enters and holds the Trojan activation state as the trigger enable signal Trig_En goes logic high. The purpose of the state machine is to ensure that the Trojan remains activated even when any of the trigger inputs changes from logic '1' back to logic '0'. This is very common when sourcing high-frequency trigger inputs, e.g., from signals on USB data buses or other high-speed input/output ports.

Fig. 9(b) and Figs. 9(e)-(f) depict three payload circuits designed to insert Trojans into commercial PCBs such as motherboards and single board computers. Fig. 9(b) is used for information leakage from any victim IC on a commercial PCB or peripheral device, and is similar to that implemented on the HaHa board. The victim IC can be any chip that transmits and/or receives data - either from other ICs on the board or from peripheral devices. Fig. 9(e) is a circuit designed to maliciously write a logic '0' into a vulnerable node of a peripheral device or an on-board IC when the trigger is enabled (i.e., Trig_En goes logic high). Finally, Fig. 9(f) uses an inverting amplifier circuit to convert positive analog voltages at the input of an on-board ADC into negative voltages, thus maliciously modifying the digitized data. As a result, the direct impact of the two latter Trojans is performance degradation of the victim device and IC. Further impacts, such as a system-wide shutdown, may occur if the device or IC is part of a critical on-board component such as the memory controller.

### C. USE CASES FOR PCB-LEVEL TROJAN EMULATION

PRISTINE can play a major role in providing assurance to PCBs that pass through an untrusted supply chain. It can be

used by PCB designers to explore viable attacks and also by system integrators or end users to verify PCB integrity. Next, we describe some possible use cases for PRISTINE.

1) Create a subset of valid attacks from possible attacks: From the possible Trojan attack space, evaluate which attacks are practically viable and rank them based on the difficulty of the triggers or the severity of the payloads.
2) Analyze feasibility of a specific attack: Carry out Trojan attacks on a commercial or custom-designed PCB and check if the attack induces any functional alteration of the system or a peripheral device or any data leakage from a specific peripheral IC on the PCB.
3) Estimate functional and side-channel (power, delay, EM, etc.) alteration due to a Trojan: Perform coarse estimation of how severe the effects of the Trojan are based on whether the system or peripheral device malfunctions, freezes, or completely shuts down. Also, estimate the amount of data leakage through the side channels and the fidelity of the leaked data.
4) Develop a quantifiable PCB assurance solution: Based on the effects of the Trojan payloads applied on a PCB, improve the PCB hardware and software design to minimize the probability of triggering the specific attack.
5) Analyse whether a Trojan is activated on a PCB through a hardware or software trigger.
6) Assess hardware Trojan detection approaches against various Trojan attacks.

## V. EXPERIMENTAL RESULTS

### A. EXPERIMENTAL SETUP AND DEMONSTRATION

#### 1) PCB TROJAN IMPLEMENTATIONS ON CUSTOM HAHA BOARDS

Several Trojans were designed to perform either malicious data writes on a HaHa board or information leakage operations using an on-board LED. Activation of these Trojans is determined by the state of four user-controlled switches on the victim HaHa board. Fig. 10(a) shows the block diagram of a Trojan inserted in the victim board that is activated when all four switches are turned on; the corresponding payload is malicious inversion of the LED state. Fig. 10(b) shows the block diagram of another Trojan that has the same trigger but a different payload, namely leakage of information on the LED state. The leaked data can be masked by adding an LFSR circuit that generates a pseudo-random code. Also, note that the AND-based trigger circuit of both these Trojans can be easily replaced by a NOR-based circuit such that the Trojans are activated only when all the switches turn off.

#### 2) PCB TROJAN IMPLEMENTATIONS ON COMMERCIAL BOARDS

In order to emulate the insertion of Trojans on commercial boards, we developed four different types of PCB Trojans and implemented them on either a motherboard or a single-board

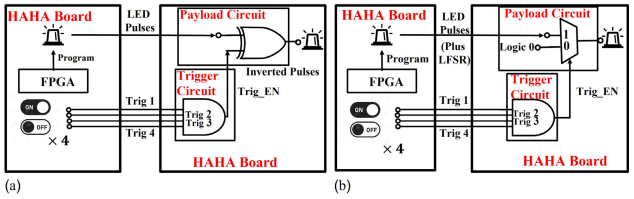


**FIGURE 10.** Block diagram of Trojans implemented on custom HaHa boards that (a) invert the state of an on-board LED, and (b) leak the state of the LED. In both cases, the Trojan is triggered when the user turns on all four on-board switches.

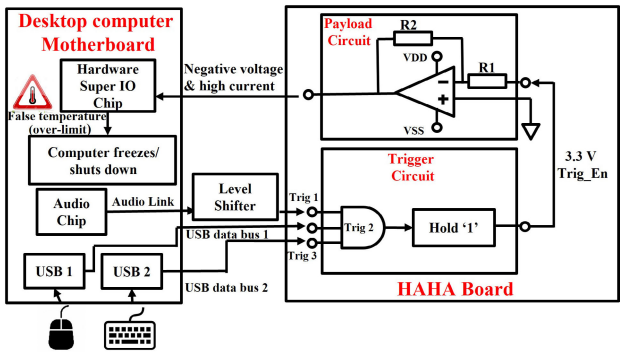


**FIGURE 11.** Block diagram of an "automatic computer freeze" Trojan triggered by a combination of USB device plug-in and audio playback events.

computer. Block diagrams of the first two designs, both of which contain a malicious data write payload, are shown in Fig. 11 and Fig. 12. As in Fig. 7, the experimental system uses a custom HaHa board as the Trojan generator and a desktop computer motherboard or a single-board computer as the victim board whose data buses are explored and modified. The first Trojan (Fig. 11) was implemented on a multi-protocol I/O and monitoring chip (F71889A from Fintek). We exploited a vulnerability in this IC's temperature monitoring function to generate false system temperature readings that can trigger unexpected system shutdowns. Specifically, the payload relies on the fact that the on-board firmware automatically freezes or shuts down the system whenever the temperature sensed by the monitoring IC rises over a certain threshold. For this purpose, the chip uses an internal ADC to digitize the analog output voltage of a board-mounted temperature sensor.

The temperature sensor voltage was found to decrease to a negative value when the temperature crosses the threshold value. Accordingly, the payload circuit uses an operational amplifier (op-amp), as shown in Fig. 9(f), to convert the 3.3 V trigger enable signal (Trig_En) to a negative voltage that emulates an above-threshold temperature reading. The analog input pin for the F71889A's temperature monitoring circuit requires a drive current of at least 0.1 A, so a high-output-current op-amp (OPA551PA, from Texas Instruments) was used to implement the payload. The trigger circuit used by this Trojan consists of a three-input AND gate followed by a simple state machine. The latter captures transient logic

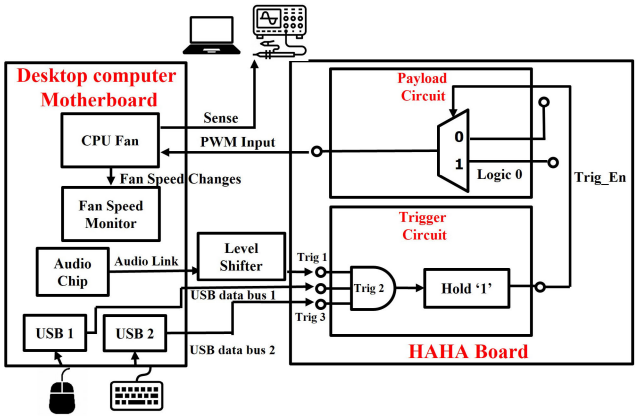


**FIGURE 12.** Block diagram of an "automatic CPU fan speed change" Trojan triggered by a combination of USB device plug-in and audio playback events.

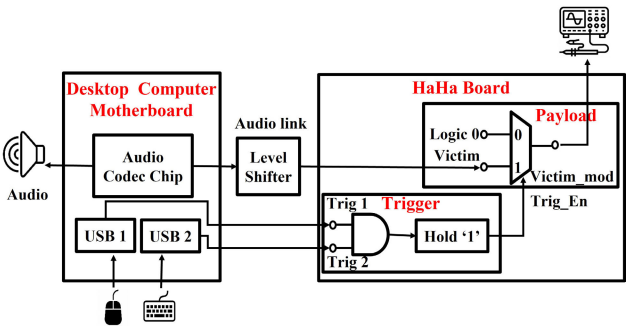


**FIGURE 13.** Block diagram of an "automatic audio data leakage" Trojan triggered by the plug-in of USB devices.

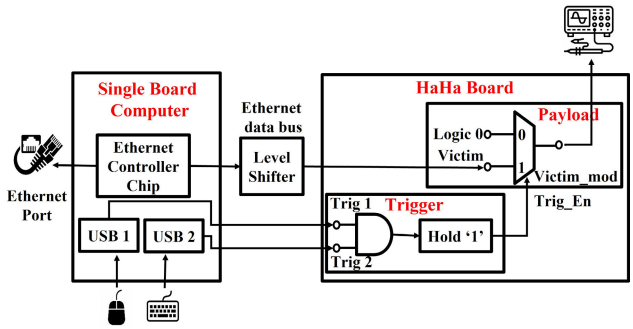


**FIGURE 14.** Block diagram of an "automatic networking data leakage" Trojan triggered by the plug-in of USB devices.

"high" events at the AND gate output and holds (i.e., latches) them to generate a digital signal (denoted by Trig_En) that permanently triggers the payload. Here, the trigger inputs are two USB devices (USB mouse and keyboard) signals and one audio output signal from an audio codec IC (ALC892, from Realtek) on the motherboard. Accordingly, the Trojan is triggered whenever the two USB devices are plugged into specific USB ports (located on the motherboard's rear panel) *and* an audio file is played for more than 3 sec. The minimum audio duration for triggering the Trojan (3 sec in this case) was optimized to eliminate the possibility of unwanted triggering due to auditory noise (e.g., short-duration sounds generated when USB devices are plugged in). In addition, a level shifter IC is used to convert the low-voltage (∼1 V) audio signals to 3.3 V logic signals that can be directly read by the FPGA.

The second Trojan was designed to maliciously modify the operation of the CPU fan, as shown in Fig. 12. The victim node is the pulse-width modulation (PWM) input pin of the four-wire CPU fan on the motherboard, which can be attacked by overriding the PWM signal with a logic '0'. This results in a change of CPU fan speed, which can be confirmed through either a hardware monitoring program or directly via the sense pin of the CPU fan connector. The payload for this Trojan is identical to that shown in Fig. 9(e). Specifically, the multiplexer only passes a logic '0' to the PWM pin when the trigger enable signal (denoted by Trig_En) is detected as a high level. The trigger circuit also uses a three-input AND gate with a state machine, as shown in Fig. 8(c). Here, the inputs are bus signals of two USB devices (USB mouse and keyboard) and audio signals from the audio IC. Thus, the impact of this Trojan is automatic and malicious change of CPU fan speed when both USB devices are plugged in and an audio file is played for more than 3 sec.

The third and fourth Trojans (shown in Fig. 13 and Fig. 14, respectively) are both focused on information leakage. The first design uses an AND-gate trigger and a multiplexer (as in Fig. 9(b)) as the payload circuit. Its goal is to leak some data from the audio bus whenever the trigger is enabled, i.e., both USB devices are physically connected. A level shifter was again used to convert low-voltage audio signals to 3.3 V. The second Trojan design (Fig. 14) leaks data bus information from an Ethernet controller IC (RTL8152B, from Realtek) on a single-board computer through a physical-layer (PHY) transmit pin in 10 Mbps mode. The RTL8152B is used for high-speed wired networking over CAT-5 UTP cable (100 Mbps) or CAT-3 UTP cable (10 Mbps).

### B. SETUP FOR EXPERIMENTAL MEASUREMENTS

A custom HaHa board (shown in Fig. 15(a)) was used to implement Trojan generation for the proposed Trojan emulation architecture (shown in Fig. 7). The board contains an ARM-based microcontroller, a low-power FPGA (Intel MAX-10 with 50K logic elements), a flash memory module, a side channel measurement module, an inertial measurement unit (IMU), a breadboard, and a few other basic elements commonly found on commercial FPGA development boards (including a seven-segment display, I/O modules, serial ports, switches, and buttons). The two primary modules used in the experiments were the FPGA chip and the breadboard; the former was programmed to implement the digital components of both trigger and payload circuits, while the latter was used to implement the analog components. The side-channel measurement module uses an instrumentation amplifier that is configured to measure the current flowing through a 1 Ω current-sensing resistor.

**TABLE 2. Summary of Trojan designs implemented on the HaHa board.**

| Trojan ID | Trigger input | Trigger circuit | Payload | Payload circuit | DUT current (mA), Trojan not activated | DUT current (mA), Trojan activated |
|---|---|---|---|---|---|---|
| Trojan 1 | 4 switches | 4-input AND gate | Invert the state of one LED on the victim board | 2-input XOR gate | 14.01 | 36.44 |
| Trojan 2 | 4 switches | 4-input NOR gate | Invert the state of one LED on the victim board | 2-input XOR gate | 5.23 | 17.86 |
| Trojan 3 | 4 switches | 4-input AND gate | Leak the state of one LED on the victim board to an LED on the Trojan generator | 2-input MUX | 4.84 | 5.73 |
| Trojan 4 | 4 switches | 4-input NOR gate | Leak the state of one LED on the victim board to an LED on the Trojan generator | 2-input MUX | 4.22 | 5.2 |
| Trojan 5 | 4 switches | 4-input AND gate | Leak the randomized state of one LED on the victim board to an LED on the Trojan generator | 2-input MUX | 21.58 | 26.82 |

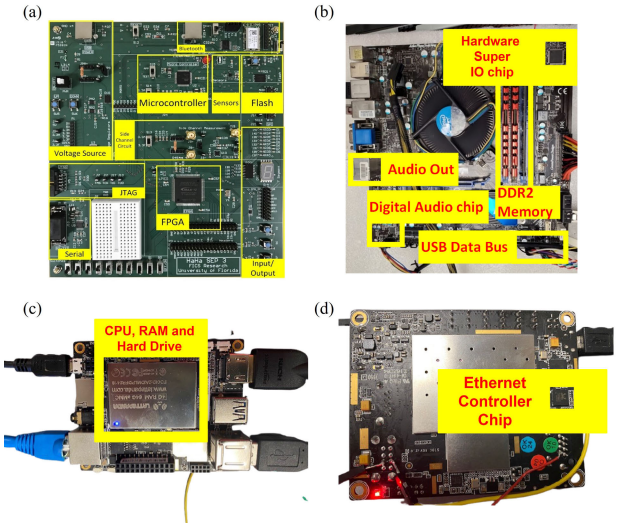


**FIGURE 15.** (a) The custom hardware security platform (HaHa board) used for building the trigger and payload circuits. (b)-(d) Trojan insertion victims: (b) a motherboard, (c)-(d) a single-board computer (both front and back sides shown).

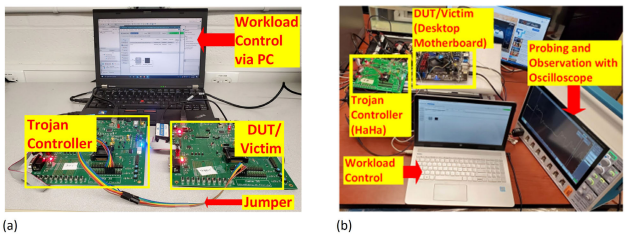


**FIGURE 16.** Experimental setup for emulating Trojan insertion using (a) the HaHa board, and (b) the motherboard.

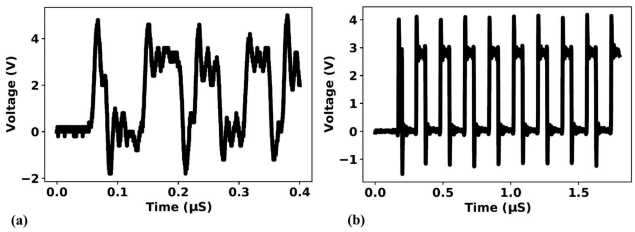


**FIGURE 17.** Typical time-domain measurement results of functional leakage-type Trojans: (a) Trojan #3, and (b) Trojan #4.

The desktop motherboard (MSI H67MA-E45-B3d with 16 GB DDR3 RAM) that was used as a Trojan insertion victim is shown in Fig. 15(b). The other victim used during the experiments was a single-board computer (LattePanda) with 4 GB RAM and 64 GB eMMC storage; its front and back sides are shown in Figs. 15(c)-(d). These two boards were chosen as being representative of older desktop computers and modern miniaturized computing platforms, respectively.

Figs. 16(a)-(b) depict the setup used for Trojan insertion experiments on the HaHa board and the motherboard, respectively. The experimental setup using the single board computer as the victim is exactly the same as for the motherboard. A second HaHa board serves as the Trojan generator if a HaHa board is chosen as the victim, as shown in Fig. 16(a). The same HaHa board also generates Trojans when the motherboard or single-board computer is chosen as the victim, as shown in Fig. 16(b). In both cases, a high-speed digital oscilloscope (1 GHz bandwidth) was used to probe waveforms on the victim board. Finally, a laptop computer was used to compile the digital Trojan circuits (defined as Verilog code) and then program the FPGA on each HaHa board.

### C. TROJAN ACTIVATION AND VALIDATION RESULTS

The setups shown in Fig. 16 were used to perform a variety of Trojan insertion experiments on both custom HaHa boards and commercial boards (motherboards and single-board computers). The results demonstrate successful Trojan insertion and payload activation, as described next.

#### 1) HARDWARE TROJAN INSERTION RESULTS ON THE HAHA BOARD

We first conducted experiments to verify the effectiveness of all five Trojans designed using the HaHa board as a

**TABLE 3.** Key metrics for hardware Trojans in commercial PCBs.

| Trigger input | Payload | Signal frequency | Probability of activation (score: 0-1) | Degree of payload impact (score: 0-5) |
|---|---|---|---|---|
| Two USB mouse signals and 3-sec audio play | Computer system freeze | DC | 0.17 | 5 |
| Two USB mouse signals and 3-sec audio play | CPU fan speed change | 35 kHz | 0.17 | 2 |
| Two USB mouse signals | Audio bus data leakage | 24 MHz | 0.25 | 4 |
| Two USB mouse signals | Ethernet controller data leakage | 10 MHz | 0.25 | 4 |

victim board, as listed in Table 2. The victim node explored during the experiments is an LED that is programmed by the workload laptop to toggle every 0.76 sec. The Trojan trigger inputs are four switches on the victim HaHa board. In each case, the final experimental results matched the desired impacts of the Trojan payload. Trojans 1 and 2 were activated when all their trigger inputs became high or low, respectively, resulting in inversion of the LED state. Trojans 3 and 4 were triggered by the same inputs to leak state information from the LED on the victim board to another LED on the Trojan generator. Trojan 5 was also verified to function as expected; the LED state information received by the Trojan generator was equivalent to the randomized information from the victim board. Table 2 lists the power consumption of the victim board (i.e., the device under test (DUT)), in two scenarios: i) when the Trojans are not activated, and ii) when they are activated. Among the five designs, Trojans 1, 2, and 5 result in a significant increase in power consumption when they are activated. As a result, these Trojans, which initiate either malicious data writes or masked information leakage, are relatively easy to detect using on-board supply current monitoring [26].

#### 2) HARDWARE TROJAN INSERTION RESULTS ON COMMERCIAL BOARDS

Next, we replaced the victim HaHa board with commercial boards (initially the motherboard and then the single-board computer). The four Trojans designed for these boards were successfully triggered and resulted in the expected payload impacts (as summarized in Figs. 11 through 14). In the case of Trojan 1, the analog voltage at the temperature sensor input pin of the hardware supervisor chip was converted from 1 V to −1 V, in turn causing the desktop computer to freeze when an audio file was played for more than 3 sec as well as a USB mouse and keyboard were plugged into selected USB ports on the back panel of the motherboard. In the case of Trojan 2, the CPU fan speed was maliciously decreased by ∼32% (from 1554 rps to 1051 rps) by triggering a frequency change of the CPU fan connector's PWM input signal (from 50 kHz to 35 kHz) when the two USB devices were plugged in. Trojans 3 and 4 were also inserted into both commercial boards and successfully acquired data bus information from the audio codec and the Ethernet controller, respectively.

The leaked data from the victim boards was successfully probed at the Trojan generator; typical time-domain measurement results are shown in Fig. 17. The signal sensed from Trojan 3 (Fig. 17(a)) is more distorted than that generated by Trojan 4 (Fig. 17(b)) due to its much higher data rate, which makes it susceptible to the effects of cable reflections.

The measurement results were analyzed to derive Trojan risk level metrics, as summarized in Table 3. Potential Trojan threats were assessed using two complementary criteria, namely probability of activation and degree of payload impact, which can be multiplied together to obtain a measure of overall risk. We quantified these criteria using scores with a range of 0 to 1 (for increasing probability of activation) and 0 to 5 (for increasing payload impact). In addition, we also specify the characteristic signal frequency (or data rate) probed by each Trojan. This is because designs that require access to higher-frequency signals are disfavored by attackers due to their increased hardware complexity and data acquisition challenges. Based on our analysis, Trojan 1 has the highest payload impact, because its activation leads to system freezes, which are very undesirable for any computing platform. This Trojan, however, is rarely triggered because its trigger conditions are more stringent than for the others (e.g., based on the number of required USB plug-in events), thus resulting in the lowest probability of activation. By contrast, Trojan 2 has the least serious payload impact, since normal system operation is largely unaffected by a modest change in CPU fan speed. Finally, Trojans 3 and 4 have higher probability of activation because of their more relaxed trigger conditions. However, they are also likely to be disfavored by attackers due to their relatively high signal frequency.

We configured the activation mechanism of each Trojan based on a trade-off between the probability of its activation and impact on the PCB system, as shown in Table 3. Generally, from the attacker's point of view, a hardware Trojan with more severe adverse impacts on the system will be designed to be less prone to activation, ensuring its stealthiness when the victim party tests the PCB during the post-fabrication phase of the PCB supply chain. On the victim's side, one suggestion to avoid the activation of Trojan 1 is to select a temperature monitor circuit that is more robust to fake above-threshold temperature readings. Regarding Trojans that involve data leakage, one approach to mitigating their serious outcomes when activated is to encrypt data communications between ICs [27].

## VI. DISCUSSION

### A. LIMITATIONS

While the Trojan emulation platform can create benchmarks for assessing and comparing various PCB Trojan countermeasures, there still exist limitations on the types of

boards and Trojans (designs, payloads, and triggers) that are supported. One limitation is that leaking sensitive information through power analysis of victim nodes on a commercial PCB is very difficult. The target signal trace path on the PCB has to be broken and a current sensor (e.g., a series resistor or Hall-effect device) added after PCB fabrication and assembly to measure current flow, which is time-consuming and demands significant PCB rework when the board has many layers. Another limitation is that signals from high-speed ICs or peripheral devices on the victim nodes cannot easily serve as Trojan triggers because of two main reasons. Firstly, the FPGA on the HaHa board uses a relatively lower clock frequency (in our case, typically set to 50 MHz) than most high-speed ICs, causing low reliability of data read/write in terms of signal/data fidelity. Secondly, high-speed I/Os based on low-voltage differential signaling (LVDS) require special impedance-matched driver circuits. In the absence of such drivers, LVDS signals above ∼100 MHz are significantly distorted after transmission through 1-2 m of cable from the victim node to either the probing system or the custom HaHa board. The resulting degradation of signal integrity causes loss of the leaked information that in turn makes the Trojan trigger unstable. A final limitation is that Trojans that maliciously write data into on-board memory may not be enabled for some ICs, such as an SPI flash memory IC with locked read/write access on a commercial motherboard or single-board computer, unless the microprocessor or BIOS chip can be reprogrammed. If such specialized knowledge of the firmware is unavailable, inserting Trojans of this type into the PCB may lead to unexpected and permanent system-level failures.

### B. FLEXIBILITY AND SCALABILITY

The proposed Trojan emulation framework can be extended to find solutions to the three aforementioned problems. Firstly, other options for side channel attacks are available, including acquiring the local magnetic field and/or infrared (thermal) profiles near working ICs and traces. For example, sensing the magnetic fields generated on the PCB with and without Trojans inserted can allow significant amounts of information to be leaked from the victim nodes. In addition, the magnetic field amplitude near a trace is proportional to current flow, which allows power consumption to be calculated. Limitations on information leakage from high-speed signals can be relaxed by replacing the FPGA on the HaHa board (i.e., the Trojan generator) with a higher-end FPGA, such as an Intel Stratix-series FPGA that runs at a fractional-$N$ phase-locked loop (PLL) output frequency up to 1 GHz, includes a 28.3 Gbps transceiver, and provides LVDS I/O modules for capturing high-speed data.

Also, the bandwidth of the high-speed links between the Trojan generator and the victim nodes can be increased by using impedance-matched connections. Specifically, for high-speed signal links, short jumper wires can be selected based on the signal wavelength, ensuring that transmission line effects will not cause signal reflections or attenuation.

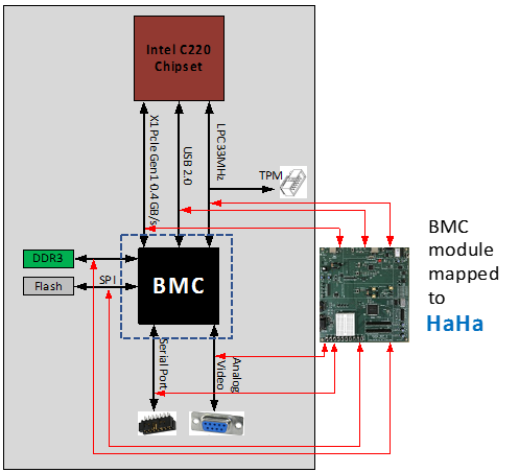


**FIGURE 18.** Implementation of the BMC module and other complex trigger logic using the FPGA present on the HaHa board.

Another method is to use 50 Ω coaxial cables to minimize impedance mismatches at high signal frequencies, thereby achieving enough bandwidth for high-speed data transmission.

In order to enable malicious data read/write from/to the memory chip, a hardware programmer or microcontroller can be connected to the chip soldered on the motherboard with enough output current to supply the power of the components that share the same power rail, allowing for re-flashing the IC for signal sniffing or tampering. Moreover, collaborative Trojans can be developed to expand the scope of the emulation platform. In this approach, a microcontroller or microprocessor on the commercial PCB can be configured to serve as either the Trojan trigger or payload, thus enabling more complex Trojans to be realized. Fig. 18 shows a possible realization of a complex Trojan using the baseboard management controller (BMC) module, as reported in [28]. The BMC module's function can be implemented into the FPGA of the HaHa board. Finally, hardware Trojan triggers or payloads can be distributed across different layers of a single PCB or among different boards within a modular PCB system, working collaboratively to activate the Trojan.

### C. INTEGRATION WITH EXISTING PCB ANALYSIS TOOLS

More measurement instruments can be integrated into the proposed Trojan emulation platform. For example, a magnetic field probe (e.g., based on Hall-effect sensors) can measure the magnetic field near traces or ICs on the victim PCB, thus enabling power-based side channel attacks as described in the previous section. Also, digital oscilloscopes or logic analyzers with higher bandwidth can be added to the setup to allow probing and decoding of high-frequency data buses associated with DDR RAM, graphics processors, and other high-speed components.

## VII. CONCLUSION

This paper has presented PRISTINE, a flexible PCB-level hardware emulation platform, which enables Trojan insertion

on both custom and commercial PCBs and observing their effects in functional as well as side-channel (e.g., supply current, timing, and EM) behavior. Hardware Trojans for both types of boards have also been specially designed and implemented. Both the Trojan triggers and payload circuits were designed for activation using either internal (i.e., on-board) or external signals. Experiments were conducted to validate Trojan insertions on various complex boards. For this purpose, an FPGA-based Trojan generator was implemented using the custom HaHa board. The resulting data was analyzed to derive benchmarks for PCB-level Trojans, thus allowing their effects to be explored without fabricating and testing a large number of board variants.

Experimental results show that Trojans triggered on HaHa boards can invert the state of an LED or leak the LED state information to the Trojan generator. In addition, Trojans inserted on motherboards and single-board computers can be triggered automatically through multiple mechanisms (e.g., via USB device plug-in events or audio signals) to achieve the desired payload effects (e.g., system freezes, malicious changes in CPU fan speed, or leakage of sensitive data from audio codecs or Ethernet controllers). Finally, the experimental results were analyzed to derive a risk metric for Trojans on commercial boards. The analysis shows that Trojans that generate the most serious outcomes (e.g., system freezes) are less prone to be triggered, thus reducing their risk. In contrast, information leakage Trojans can be activated more often but have less impact on the system. We expect these conclusions to be helpful for system designers by guiding their adoption of countermeasures for different types of PCB Trojans. Future work will focus on further improvement in the scalability and flexibility of the PRISTINE platform.

## REFERENCES

[1] S. Bhunia and M. Tehranipoor, *The Hardware Trojan War*. Cham, Switzerland: Springer, 2018.

[2] F. Khalid, S. R. Hasan, O. Hasan, and F. Awwad, "Runtime hardware trojan monitors through modeling burst mode communication using formal verification," *Integration*, vol. 61, pp. 62–76, Mar. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167926016301535

[3] S. R. Hasan, S. F. Mossa, O. S. A. Elkeelany, and F. Awwad, "Tenacious hardware trojans due to high temperature in middle tiers of 3-D ICs," in *Proc. IEEE 58th Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Aug. 2015, pp. 1–4.

[4] S. H. Russ and J. Gatlin, "Ways to hack a printed circuit board: PCB production is an underappreciated vulnerability in the global supply chain," *IEEE Spectr.*, vol. 57, no. 9, pp. 38–43, Sep. 2020.

[5] J. Harrison, N. Asadizanjani, and M. Tehranipoor, "On malicious implants in PCBs throughout the supply chain," *Integration*, vol. 79, pp. 12–22, Jul. 2021.

[6] D. Mehta, H. Lu, O. P. Paradis, M. T. Rahman, Y. Iskander, P. Chawla, D. L. Woodard, M. Tehranipoor, and N. Asadizanjani, "The big hack explained," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 16, no. 4, pp. 1–25, Oct. 2020. [Online]. Available: https://api.semanticscholar.org/CorpusID:222111388

[7] C. Vaughan, "Xbox security issues and forensic recovery methodology (utilising Linux)," *Digit. Invest.*, vol. 1, no. 3, pp. 165–172, Sep. 2004.

[8] S. Bhunia and M. Tehranipoor, *Hardware Security: A Hands-on Learning Approach*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2018.

[9] K. Rosenfeld and R. Karri, "Attacks and defenses for JTAG," *IEEE Des. Test Comput.*, vol. 27, no. 1, pp. 36–47, Jul. 2010.

[10] S. Yang, S. D. Paul, and S. Bhunia, "Hands-on learning of hardware and systems security," *Adv. Eng. Educ.*, vol. 9, no. 2, p. n2, 2021.

[11] S. Bhunia and M. Tehranipoor, *Hardware Security: A Hands-on Learning Approach*. San Mateo, CA, USA: Morgan Kaufmann, 2018.

[12] T. Hoque, S. Yang, A. Bhattacharyay, J. Cruz, and S. Bhunia, "An automated framework for board-level trojan benchmarking," 2020, *arXiv:2003.12632*.

[13] S. Ghosh, A. Basak, and S. Bhunia, "How secure are printed circuit boards against trojan attacks?" *IEEE Des. Test. Comput.*, vol. 32, no. 2, pp. 7–16, Apr. 2015.

[14] B. Shakya, T. He, H. Salmani, D. Forte, S. Bhunia, and M. Tehranipoor, "Benchmarking of hardware trojans and maliciously affected circuits," *J. Hardw. Syst. Secur.*, vol. 1, no. 1, pp. 85–102, Mar. 2017.

[15] S. Paley, T. Hoque, and S. Bhunia, "Active protection against PCB physical tampering," in *Proc. 17th Int. Symp. Quality Electron. Design (ISQED)*, Mar. 2016, pp. 356–361.

[16] J. Cruz, Y. Huang, P. Mishra, and S. Bhunia, "An automated configurable trojan insertion framework for dynamic trust benchmarks," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2018, pp. 1598–1603.

[17] *Analog/Mixed-Signal Simulation*. Accessed: Mar. 29, 2024. [Online]. Available: https://https://www.cadence.com/en_US/home/tools/pcb-design-and-analysis/analog-mixed-signal-simulation.html

[18] *TINA-TI*. Accessed: Mar. 29, 2024. [Online]. Available: https://www.ti.com/tool/TINA-TI

[19] T. F. Wu, K. Ganesan, Y. A. Hu, H.-S. P. Wong, S. Wong, and S. Mitra, "TPAD: Hardware trojan prevention and detection for trusted integrated circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 4, pp. 521–534, Apr. 2016.

[20] C. Bolchini, L. Cassano, I. Montalbano, G. Repole, A. Zanetti, and G. D. Natale, "HATE: A hardware trojan emulation environment for microprocessor-based systems," in *Proc. IEEE 25th Int. Symp. On-Line Test. Robust Syst. Design (IOLTS)*, Jul. 2019, pp. 109–114.

[21] H. Pearce, V. R. Surabhi, P. Krishnamurthy, J. Trujillo, R. Karri, and F. Khorrami, "Detecting hardware trojans in PCBs using side channel loopbacks," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 30, no. 7, pp. 926–937, Jul. 2022.

[22] *Palladium Emulation*. Accessed: Dec. 20, 2023. [Online]. Available: https://www.cadence.com/en_U.S./home/tools/system-design-and-verification/emulation-and-prototyping/palladium.html

[23] *Zebu Emulation*. Accessed: Dec. 20, 2023. [Online]. Available: https://www.synopsys.com/verification/emulation.html

[24] *Veloce Strato and Veloce Strato+ Hardware*. Accessed: Dec. 20, 2023. [Online]. Available: https://eda.sw.siemens.com/en-U.S./ic/veloce/strato-hardware/

[25] M. McGuire, U. Ogras, and S. Ozev, "PCB hardware trojans: Attack modes and detection strategies," in *Proc. IEEE 37th VLSI Test Symp. (VTS)*, Apr. 2019, pp. 1–6.

[26] 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. 740–751, Jun. 2022.

[27] Z. Guo, X. Xu, M. M. Tehranipoor, and D. Forte, "EOP: An encryption-obfuscation solution for protecting PCBs against tampering and reverse engineering," 2019, *arXiv:1904.09516*.

[28] J. Robertson and M. Riley, "The big hack: How China used a tiny chip to infiltrate U.S. companies," *Bloomberg Businessweek*, 2018.

**JUNJUN HUAN** received the B.S. degree in electronic information engineering and the M.S. degree in electrical engineering from the University of Dayton, Dayton, OH, USA. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Florida, Gainesville, FL, USA. His current research interests include electronics for ultrasound imaging and image-guided treatment, wearable medical sensors, and hardware security.

**SHUBHRA DEB PAUL** (Member, IEEE) received the B.Sc. degree in electrical and electronic engineering from Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2012, the M.S. degree in electrical engineering from Oklahoma State University, Stillwater, OK, USA, in 2016, and the Ph.D. degree in electrical and computer engineering from the University of Florida, Gainesville, FL, USA, in 2022. His research interests include security and trust, digital circuit simulation, and digital and embedded systems.

**SOUMYAJIT MANDAL** (Senior Member, IEEE) received the B.Tech. degree from Indian Institute of Technology (IIT) Kharagpur, India, in 2002, and the S.M. and Ph.D. degrees in electrical engineering from Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 2004 and 2009, respectively. He was a Research Scientist with SchlumbergerDoll Research, Cambridge (2010–2014); an Assistant Professor with the Department of Electrical Engineering and Computer Science, Case Western Reserve University, Cleveland, OH, USA (2014–2019); and an Associate Professor with the Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL, USA (2019–2021). He is currently a Research Staff Member with the Instrumentation Division, Brookhaven National Laboratory, Upton, NY, USA. He has over 175 publications in peer-reviewed journals and conferences and has been awarded 26 patents. His research interests include analog and biological computation, magnetic resonance sensors, low-power analog and RF circuits, and precision instrumentation for various biomedical and sensor interface applications. He was a recipient of the President of India Gold Medal, in 2002, the MIT Microsystems Technology Laboratories (MTL) Doctoral Dissertation Award, in 2009, the T. Keith Glennan Fellowship, in 2016, and the IIT Kharagpur Young Alumni Achiever Award, in 2018.

**SWARUP BHUNIA** (Fellow, IEEE) received the B.E. (Hons.) from Jadavpur University, Kolkata, India, the M.Tech. degree from Indian Institute of Technology (IIT), Kharagpur, and the Ph.D. degree from Purdue University, IN, USA. Currently, he is a Professor and the Semmoto Endowed Chair of the University of Florida, FL, USA. Earlier, he was appointed as the T. and A. Schroeder Associate Professor in electrical engineering and computer science with Case Western Reserve University, Cleveland, OH, USA. He has over ten years of research and development experience with over 200 publications in peer-reviewed journals and premier conferences. His research interests include hardware security and trust, adaptive nanocomputing, and novel test methodologies. He received IBM Faculty Award, in 2013, National Science Foundation CAREER Award, in 2011, Semiconductor Research Corporation Inventor Recognition Award, in 2009, SRC Technical Excellence Award as a Team Member, in 2005, and several best paper awards/nominations. He has been serving as an Associate Editor for IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, IEEE TRANSACTIONS ON MULTI-SCALE COMPUTING SYSTEMS, and *ACM Journal of Emerging Technologies*; and served as a Guest Editor for IEEE Design & Test of Computers (2010 and 2013) and IEEE JOURNAL ON EMERGING AND SELECTED TOPICS IN CIRCUITS AND SYSTEMS (2014). He has served in the organizing and program committee for many IEEE/ACM conferences.