INVESTIGATION INTO DETECTION OF HARDWARE TROJANS ON PRINTED CIRCUIT BOARDS

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of
Doctor of Philosophy

by

Gor Piliposyan

Department of Electrical Engineering and Electronics
Faculty of Science and Engineering
March 2023

Declaration

This work has not been submitted in substance for any other degree or award at this or any other university or place of learning, nor is being submitted concurrently in candidature for any degree or other award. This thesis is the result of my own independent work/investigation, except where otherwise stated. Other sources are acknowledged by explicit references. I hereby give consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed: Gor Piliposyan

March 2023

Abstract

The modern semiconductor device manufacturing flow is becoming increasingly vulnerable to malicious implants called Hardware Trojans (HT). With HTs becoming stealthier a need for more accurate and efficient detection methods is becoming increasingly crucial at both Integrated Circuit (IC) and Printed Circuit Board (PCB) levels. While HT detection at an IC level has been widely studied, there is still very limited research on detecting and preventing HTs implanted on PCBs. In recent years the rise of outsourcing design and fabrication of electronics, including PCBs, to third parties has dramatically increased the possibility of malicious alteration and consequently the security risk for systems incorporating PCBs. Providing mechanical support for the electrical interconnections between different components, PCBs are an important part of electronic systems. Modern, complex and highly integrated designs may contain up to thirty layers, with concealed micro-vias and embedded passive components. An adversary can aim to modify the PCB design by tampering the copper interconnections or inserting extra components in an internal layer of a multi-layer board. Similar to its IC counterpart, a PCB HT can, among other things, cause system failure or leakage of private information. The disruptive actions of a carefully designed HT attack can have catastrophic implications and should therefore be taken seriously by industry, academia and the government.

This thesis gives an account of work carried out in three projects concerned with HT detection on a PCB. In the first contribution a power analysis method is proposed for detecting HT components, implanted on the surface or otherwise, consuming power from the power distribution network. The assumption is that any HT device actively tampering or eavesdropping on the signals in the PCB circuit will consume electrical power. Harvesting this side-channel effect and observing the fluctuations of power consumption on the PCB power distribution network enables evincing the HT. Using a purpose-built PCB prototype, an experimental setup is developed for verification of the methodology. The results confirm the ability to detect alien components on a PCB without interference with its main functionality.

In the second contribution, the monitoring methodology is further developed by applying machine learning (ML) techniques to detect stealthier HTs, consuming power from I/O ports of legitimate ICs on the PCB. Two algorithms, One-Class Support Vector Machine (SVM) and Local Outlier Factor (LOF), are implemented on the legitimate power consumption data harvested experimentally from the PCB prototype. Simulation results are validated through real-life measurements and experiments are carried out on the prototype PCB. For validation of the ML classification models, one hundred categories of HTs are modelled and inserted into the datasets. Simulation results show that using the proposed methodology an HT can be detected with high prediction accuracy (F1-score at 99% for a 15 mW HT). Further, the developed ML model is uploaded to the prototype PCB for experimental validation. The results show consistency between simulations and experiments, with an average discrepancy of $\pm 5.9\%$ observed between One-Class SVM simulations and real-life experiments. The machine learning models developed for HT detection are low-cost in terms of memory (around 27 KB).

In the third contribution, an automated visual inspection methodology is proposed for detecting HTs on the surface of a PCB. It is based on a combination of conventional computer vision techniques and a dual tower Siamese Neural Network (SNN), modelled in a three-stage pipeline. In the interest of making the proposed methodology broadly applicable a particular emphasis is made on the imaging modality of choice, whereby a regular digital optical camera is chosen. The dataset of PCB images is developed in a controlled environment of a photographic tent. The novelty in this work is that, instead of a generic production fault detection, the algorithm is optimised and trained specifically for implanted HT component detection on a PCB, be it active or passive. The proposed HT detection methodology is trained and tested with three groups of HTs, categorised based on their surface area, ranging from 4 mm^2 to 280 mm^2 and above. The results show that it is possible to reach effective detection accuracy of 95.1% for HTs as small as 4 mm^2. In case of HTs with surface area larger than 280 mm^2 the detection accuracy is around 96.1%, while the average performance across all HT groups is 95.6%.

Acknowledgements

For many years prior to starting my PhD studies, before I had even set my foot in any university, I remember I had made up my mind about getting a PhD degree. Many times I was told that it is not an easy journey. That successfully completing a PhD degree will take a lot of effort. Of course, other than those verbal warnings, I had no idea what it would actually take. Now, I know! Exactly four years and three months after I first started working towards my PhD degree, I know what a formidable project it is. It is safe to say that without all the support and encouragement I have received from people around me this colossal endeavour would have been extremely difficult to undertake, if not impossible. I would like to individually thank those people.

First and foremost, I would like to convey my words of gratitude to Dr. Saqib Khursheed, my primary supervisor and, at this point, my senior friend. I thank him for his constant guidance and encouragement. I have thoroughly enjoyed all our discussions on the topics of science, research and life as a whole. I thank him for always setting higher bars and pushing me to reach them. For these and all the other things unsaid, I thank you. Secondly, I would like to thank Prof. Steve Hall and Dr. Harm Van Zalinge for their advices, encouragement and constructive criticism I have received on my research along the years. I would also like to thank Dr. Daniele Rossi for his contributions in editing one of the published papers of this research and for his academic support.

I would also like to thank my dear aunt Dr. Gayane Piliposyan and uncle Prof. Peter Appleby, who have gone above and beyond in their support. For all the discussions and encouragement, for sharing your experience and hours spent proofreading my papers, for taking me out for classical concerts, walks in the Welsh countryside and the tour in Scotland when I was going through tough times, I thank you both for all that.

I have also enjoyed being around and working with my friends Dr. Antonio Leonel Hernandez Martinez, Mr. Turki Alnuayri and Mr. Anuraag Narang. For the positive research environment they had created and the discussions we had, I thank them. It has been my pleasure to work alongside with them.

Last but not least, I would like to thank my cousin Dr. Davit Piliposyan and his friend Dr. Davit Aznaurov for all the brief yet insightful discussions we have had. Their input in the search for solution paths in my research has been crucial and considerable. For all their advices and time, I would like to convey my words of gratitude.

Finally, my PhD research would have not been possible if not for the support from the department of Electrical Engineering and Electronics, University of Liverpool, UK. Besides the financial support from the department, I would also like to acknowledge its members of student support staff, who have always been there to help me and guide through problems throughout these years.

Contents

List of Figures

List of Tables

Acronyms

AQA	Automated Quality Assurance
AVI	Automated Visual Inspection
BJT	Bipolar Junction Transistor
CAD	Computer Aided Design
CNN	Convolutional Neural Network
DPM	Differential Power Monitoring
DT	Decision Tree
DTL	Diode-Transistor Logic
FPGA	Field Programmable Gate Array
FPR	False Positive Rate
GM	Golden Model
HT	Hardware Trojan
I^2C	Inter-Integrated Circuit
IC	Integrated Circuit
ICT	In-circuit Testing
IP	Intellectual Property
JTAG	Joint Test Action Group
KNN	K-Nearest Neighbour
LOF	Local Outlier Factor
MB	Monitoring Block
MCU	Microcontroller Unit
ML	Machine Learning

MPM	Main Power Monitor
NPN	Negative-Positive-Negative
PCB	Printed Circuit Board
PUI	PCB Under Inspection
RANSAC	Random Sample Consensus
RBF	Radial Basis Function
RE	Reverse Engineering
REI	Reverse Engineering Improvement
RTL	Resistor-Transistor Logic
SMC	Super Micro Computer
SNN	Siamese Neural Network
SPI	Serial Peripheral Interface
SPM	Sub Power Monitors
SVM	Support Vector Machine
TPM	Trusted Platform Module
TTL	Transistor-Transistor Logic

Chapter 1

Introduction

1.1 Motivation

Printed Circuit Boards (PCBs) are core parts of electronic devices used across a wide range of industries including consumer electronics (smartphones, computers etc), medical devices, automotive components (navigation, control systems and sensors), telecommunications equipment and military applications. In recent years companies often outsource the production of PCBs in order to decrease manufacturing costs, reduce the time-to-market, and mitigate market supply shortages. This production mode inherently introduces security risks by granting third parties access to the manufacturing lifecycle, thus giving rise to the possible introduction of malicious alterations or inclusions, such as hardware Trojans (HT). The altered hardware can give an adversary unauthorised access to the device and initiate retrieval or corruption of classified information. The destructive activities of HTs can cause catastrophic consequences including paralysing major financial or military systems, shortening operational lifetime of hardware or initiating complete failure of the system [1].

A report on hardware Trojan attacks on PCBs was published by Bloomberg in 2018 [2]. The report described a formidable scale tampering attack on company-private server infrastructures with a tiny malicious chip which, albeit carrying a small logic, tricked the motherboards providing backdoor access when the server booted up. According to the article, about thirty companies downstream the supply chain from Super Micro Computer (SMC) company were heavily affected including Amazon, Apple as well as a few government contractors. Although the SMC company swiftly denied any adversarial infiltration into their products during the manufacturing process [3], it has been since demonstrated [4] that such attacks are realistic and plausible. Possible security threats include tampering,

Trojan insertion, cloning and other attacks which can cause malfunctions such as denial of service, information leakage and backdoor access to adversaries. The Bloomberg article was later analysed in [5] where the vulnerabilities in modern PCB supply chains and existing viable countermeasures were discussed.

The attack described by Bloomberg in [2] is a typical Trojan attack when the adversaries infiltrated into the system by malicious alterations and inclusions [6], [7], [8]. Modern PCBs are complex structures with up to thirty layers, embedded passive components and hidden vias [9]. All these provide the adversaries with better opportunities to disguise an HT within a PCB, for example by tampering the design by changing the interconnections or fitting alien components in one of internal layers of the board [10], [11]. Board level Trojan attacks can occur during design, manufacturing and post-manufacturing transportation stages. For example, these attacks can change the width and spacing of PCB traces during the design stage, causing modifications in delay and cross-talk parameters, leading to parametric failure of the system [10]. Traces can be altered triggering overheating, critical delay failures or electromigratory failures in microscopic wires. Further, the value of passive components or their functionality can be modified, causing breakdown under specific operating conditions [12].

Ideally any malicious modification within an integrated circuit (IC) or PCB should be detected during pre-silicon or pre-production and post-production verification and testing stages. However, golden models are needed for pre-silicon verification, which is not always available. In addition, modern sophisticated multi-module IC and PCB designs are not always easy to verify. Verification at post-silicon stage can be done either by reverse engineering, destructive depackaging [13] or by comparing the characteristics with a golden model [14], [15], [16]. However, since the complexity of modern ICs and PCBs continues to grow and the HTs become increasingly stealthier, with a wider range of possible attacks, neither destructive verification nor traditional post-manufacturing logic testing are suitable for assured HT detection. That is why research on developing methodologies for new and effective countermeasures against HTs is crucial.

1.2 Scope

Malicious HT attacks have been reported at higher levels of system abstraction such as PCBs, which are becoming increasingly exposed and vulnerable to unwanted modifications during design or fabrication in untrusted facilities [6], [7], [8]. An attacker can try to modify the PCB design by tampering the interconnections or inserting extra components in an internal layer of a multi-layer board [10].

While HT attacks on ICs have been investigated extensively since 2007, malicious modifications to PCBs started attracting attention only in 2015 [10]. Much of the research on detection and classification of HTs in PCBs has been carried out based on understanding of IC-level HTs. However, there are key differences between IC and PCB level Trojans in several aspects, which determine the requirement of new research approaches for finding countermeasures against PCB level Trojans.

Firstly, post-manufacturing modification is normally not included in IC HT threat models, but is common in PCB level Trojan attacks. There are many cases of real life in-field attacks on PCBs, examples including malicious modifications to the PCBs in fuel pump controllers [17] and nation-state spy inclusions [18].

The attack surface too is different between IC and PCB level HTs. PCB Trojans in general communicate with functional blocks, whereas IC level HTs change the performance of these blocks. For instance, PCB HTs aiming to target trusted platform modules (TPMs) cause vulnerable communication protocols to bypass the root of trust of the TPM [19]. Whereas an IC HT would add logic in a TPM and, for example, reveal keys when triggered.

Lastly, countermeasures against PCB HTs are affected by the larger size of PCBs. If, for example, optical-based countermeasures can effectively use cameras for PCBs, imaging ICs would be more problematic requiring nanometer resolution and more advanced equipment for failure analysis.

Many countermeasure methods developed against IC level Trojans can be used for detecting and preventing PCB level HTs. However, even these methods need to be adapted to be applied on PCBs [20]. For example, countermeasures based on side-channel analysis and physically unclonable functions (PUFs) would require an adaptation to electrical

parameters of PCBs, and design for security methods should consider diverse ICs and specific discrete passive components [20]. Due to the differences between PCB and IC Trojans the supply chain and post-manufacturing attacks can also be different, requiring new approaches in board-level HT countermeasures [20].

The overall goal of the research in this thesis is to develop novel approaches for PCB Trojan detection. Specifically, three PCB-level Trojan detection problems have been targeted in three separate projects, constituting Chapters 3, 4 and 5 of this thesis. Each project has a distinct aim and a proposed approach for reaching the solution. The first two projects, in Chapters 3 and 4, should be viewed as two parts of a single larger system, both using the same circuit architecture to perform power analysis for HT detection. The first approach is based on a technique referred to as differential power monitoring, while the second makes use of machine learning algorithms. The aim of the third project in Chapter 5, on the other hand, is to detect HTs on the surface of the PCB using optical digital images for automated visual inspection. The third approach has been developed by combining conventional computer vision and deep learning techniques.

1.3 Contributions

1. Differential power monitoring method: The first contribution is a power monitoring based method. The name of the method is derived from the fact that it is seeking to detect the difference between expected and actual power consumption on the PCB, targeting the PCB's power distribution network (PDN) for monitoring. The idea is that at all times the power consumption measured on the input of the PCB should closely match the combined power consumption of the components installed on it, minus any natural power loss induced by the parasitics on the PDN. In case of a notable change between the two values, it can be assumed that there is an extra unlicensed component operating on the board, i.e. a hardware Trojan. The method is experimentally confirmed to be functioning, capable of detecting HT implants, while having no impact on the intended operations on the PCB.

2. Machine learning based method: The second contribution reuses the monitoring circuit architecture proposed in the first contribution to reduce the added overheads to none, while significantly increasing the monitoring capabilities. The method applies ML algorithms on the legitimate power consumption data (i.e. from HT clean PCB) collected experimentally from the prototype PCB. This project targeted HT devices, which consume power from a chosen legitimate IC on the PCB by connecting to one of its I/O ports.

Two ML algorithms, One-Class Support Vector Machine (SVM) and Local Outlier Factor (LOF), have been used for the simulations. One-Class SVM has been selected to be uploaded on the prototype PCB for further experimental validation. The final contribution produced in this project is a single unified ML model, aware of correct power consumption behaviour of all blocks of the PCB, capable of performing in-field real time monitoring. The experiments confirm that both algorithms show a good performance, in some cases reaching over 99% F1-score classification results. The model developed in this contribution for the prototype PCB is light weight, consuming only 27 KB of memory and the run time spatial complexity of One-Class SVM is in the order of $O(n^2)$.

3. Computer vision based method: The third contribution proposes a method for inspecting PCBs through digital optical image. Shifting attention from power analysis methods proposed in the first two contributions, the third project makes use of computer vision techniques to locate malicious inclusions on the surface of the PCB. Introducing a three stage algorithm pipeline, this project involves image processing techniques such as applying a filter, pixel-wise subtraction of two images, as well as more advanced techniques such as convolutions and deep neural networks. Tested on a range of different HT types and sizes, this contribution introduces how a combination of separately existing techniques can allow detection of unauthorised inclusions with over 95% accuracy.

1.4 Thesis Outline

The thesis is organised as follows:

- **Chapter 2:** Provides background information on HTs. The vulnerabilities of the PCB supply chain have been discussed with possible Trojan attacks. PCB level Trojans and their functionality have also been presented with examples of functional and parametric HTs. Further, board level Trojan taxonomy has been presented and its importance in identifying Trojans according to their characteristics discussed. This is followed by the description of publicly available Trojan benchmarks of PCB HTs. The rest of Chapter 2 provides a literature review of research on countermeasures of IC level Trojans followed by the review of countermeasures of PCB level HTs. Research on HT detection and prevention is categorised based on the presented taxonomy on countermeasures against HTs.

- **Chapter 3:** Proposes a PCB level HT detection method based on differential power monitoring. Using a purpose-built PCB hardware prototype, an experimental setup has been designed and fabricated to be tested on, for verification of the proposed methodology. The results confirm the ability to detect alien components on the PCB and provide directions for further research.

- **Chapter 4:** Introduces a machine learning (ML) based approach for run-time Trojan detection on the PCB. Two ML methods have been applied to detect HTs operating on power drawn from I/Os of legitimate chips on the PCB. The PCB prototype developed for the first project has been reused to obtain real-life silicon data on power consumption patterns on the PCB. The data has later been used to train two ML algorithms: One-Class Support Vector Machine and Local Outlier Factor. For validation of the ML classifiers, one hundred categories of HT devices have been modelled and inserted into the validation and testing datasets. Further, the ML model has been uploaded to the prototype PCB for hard-silicon validation of the proposed methodology.

- **Chapter 5:** Develops an automated visual inspection methodology for detecting HTs

on a PCB, using input data from a low-cost digital optical camera. It is based on a combination of conventional computer vision techniques and a dual tower Siamese Neural Network (SNN), designed in a three stage pipeline. Further, a dataset of PCB images has been developed in a controlled environment of a photographic tent.

- **Chapter 6:** Summarises the main results of all projects outlined in this thesis and outlines directions of future work.

1.5 Publications

The work which constitutes this thesis is based on the following contributions that have been published.

1. G. Piliposyan, S. Khursheed, and D. Rossi, (2022) "Hardware Trojan detection on a PCB through differential power monitoring," IEEE Transactions on Emerging Topics in Computing, Vol. 10(2), pp. 740– 751 (Number of Google Scholar citations 13). (doi: 10.1109/TETC.2020.3035521)
2. G. Piliposyan and S. Khursheed, (2022), "PCB Hardware Trojan Run-Time Detection Through Machine Learning," IEEE Transactions on Computers. (doi: 10.1109/TC.2022.3230877)
3. G. Piliposyan and S. Khursheed, (2022) "Computer Vision for Hardware Trojan Detection on a PCB Using Siamese Neural Network," IEEE International Conference on Physical Assurance and Inspection of Electronics (PAINE), Huntsville, AL, USA, October 2022, pp.1-7. (doi: 10.1109/PAINE56030.2022.10014967)

Awards and Prizes

1. Awarded First Prize and Golden Medal in the Engineering Category in the Parliamentary and Scientific Committee's STEM for Britain 2022 competition, London, March 2022. http://www.stemforbritain.org.uk/2022-winners/
2. Awarded IEEE Communications Society, Distinction in Research Subject Communication Skills in STEM for Britain 2022, London, March 2022.

Chapter 2

Printed Circuit Board Hardware Trojan Preliminaries

2.1 Introduction

A hardware Trojan (HT) is a rogue piece of extra hardware or an alteration of the existing PCB layout without adding extra components, secretly implemented by adversaries for malevolent reasons such as information gathering, false signalling, hijacking control, etc. Trojans can be inserted into an integrated circuit (IC) or a printed circuit board (PCB) [21], [22] and give an attacker unauthorised access into the device and, for example, initiate a leakage or corruption of important information [10]. In a business world, Trojans can be designed with the intention of harming, or even destroying, the reputation of a rival company [23].

PCB Trojan attacks can happen during design or manufacturing stages in an untrusted design house or foundry. Adversaries try to design the Trojans to be as stealthy as possible, in order to evade detection during post-manufacturing tests.

2.2 Printed Circuit Board Supply Chain Vulnerabilities

The rise of outsourcing of PCB design and fabrication process to third parties in recent years has created various vulnerabilities and threats in the supply chain. The stages in PCB production and supply chain and some of the existing vulnerabilities are shown in Figure 2.1 [24], [20].

PCB design starts by defining its specifications and converting them into schematics and board layouts, which are then exported and sent to foundries. In the PCB fabrication

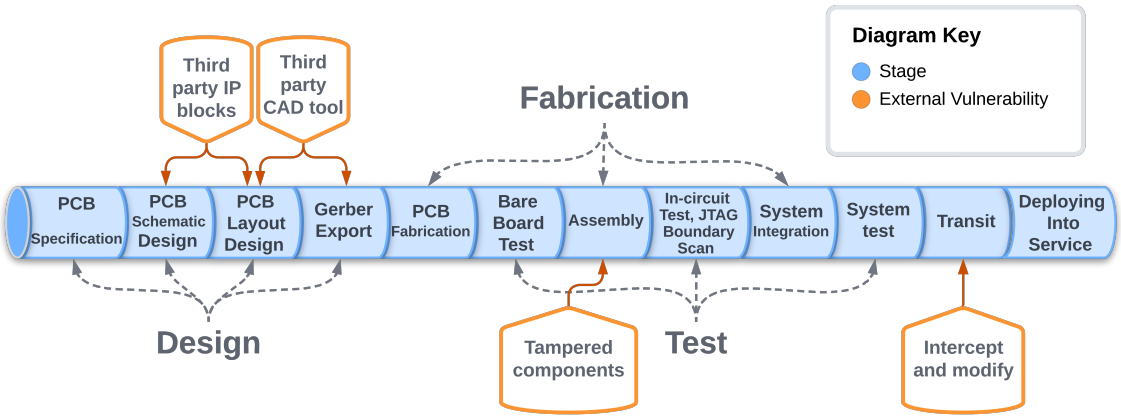


Figure 2.1: Vulnerabilities in IC and PCB supply chain and production phase.

stage, the board substrate is built by bonding several layers of conductive foil with epoxy. This is followed by defining traces through chemical etching before connecting different layers through drilling and plating vias. At the assembly stage the bare board is populated with corresponding components and after passing through in-circuit tests and JTAG boundary scans the assembled board is incorporated in a larger mechanical or electrical system during the system integration stage. Then the completed final system is tested and shipped.

The vulnerabilities can be exploited by adversaries at any point of the PCB supply chain. Possible attacks on PCBs can be categorised as hardware Trojan insertion, piracy and counterfeiting, and in-field alteration, shown in Figure 2.2. The attackers can change both the width and spacing of board traces, modify the delay parameters and cause parametric failures during manufacturing stage or in the field [10]. They can also modify traces and create sources of delayed electromigratory failures in the future, which may not be detected by parametric tests at the time of fabrication. Finally, the adversaries can change the functionality of passive components and cause malfunction, shortened operational lifetime or denial of service.

Pirated PCBs can be used to clone and overproduce PCBs, which will harm companies that put significant investments in research and development. In addition, cloned PCBs, having unlikely been rigorously tested, would be less reliable compared to their genuine counterparts and could cause fatal problems if used in security-sensitive and critical infrastructures. Further, with Trojan insertion adversaries can get a back-door access

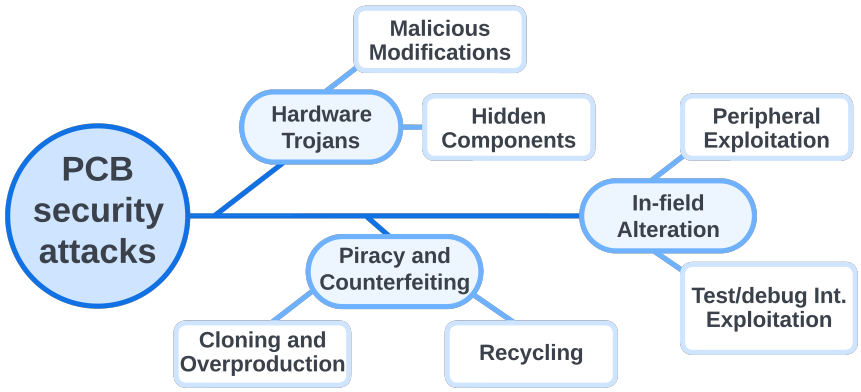


Figure 2.2: General overview of IC and PCB related malicious activities.

to carry out a stealthy attack with potential catastrophic consequences. Therefore, it is crucial to authenticate PCBs before they are deployed for service [24].

2.3 Printed Circuit Board Level Hardware Trojans

In general, a Trojan has an optional activation mechanism referred to as a trigger and an objective purpose referred to as payload [25]. A simplified diagram of an HT is shown in Figure 2.3, where the trigger of the Trojan activates the payload causing a failure by altering signal S to S_0 [21]. The trigger monitors different signals and/or various functions in the system. The payload connects to the output of the trigger and signal S from the original circuit. When the trigger identifies a specified signal or activation condition, the payload is activated to deliver the malicious action. The payload is mostly inactive since the trigger is designed to activate under rare conditions in order for the HT to remain undetected. Therefore most of the time during design and testing stages the system operates as Trojan-free [25], making detection of the HT more difficult.

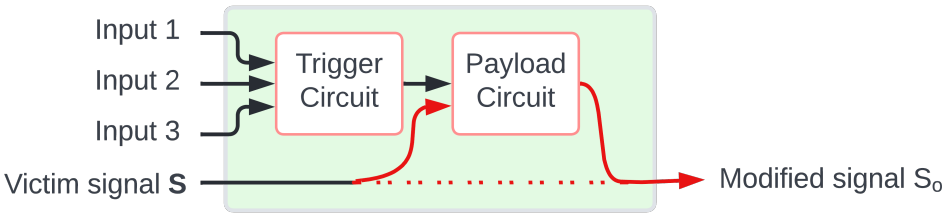


Figure 2.3: Generic block diagram structure of a hardware Trojan.

Modern PCBs contain multiple microelectronic components including simple passive devices (diodes, transistors, resistors, capacitors) and diverse ICs (microprocessors, FPGAs, memories). An adversary can design a PCB level HT by implanting extra components. These types of Trojans are referred to as functional. For example, a number of extra NPN bipolar junction transistors (BJTs) can be introduced, which would operate and look like existing legitimate NPN BJTs on the PCB. However, instead of supporting the planned operation of the PCB they would rather have some adversarial effect.

PCB-level Trojan attacks could also be carried out without introducing additional components, by only changing the structural and/or parametric properties of conductive and non-conductive parts of the PCB layers. These types of Trojans are called parametric. Trojan attacks involving intra-component or board level alterations would also be possible in more complex scenarios, where a combination of several types of HTs could be introduced to the circuit.

2.3.1 Functional Trojan

The general structure of a functional Trojan with a trigger and a payload, is shown in Figure 2.4. The trigger receives one or several signals from existing PCB traces to develop a rare-activation trigger condition [26], necessary for minimising the possibility of Trojan activation during functional testing of the PCB. This can be achieved by incorporating a large number of traces which rarely meet an activation condition or by building a trigger circuit which needs more than one excitation to meet the activation condition.

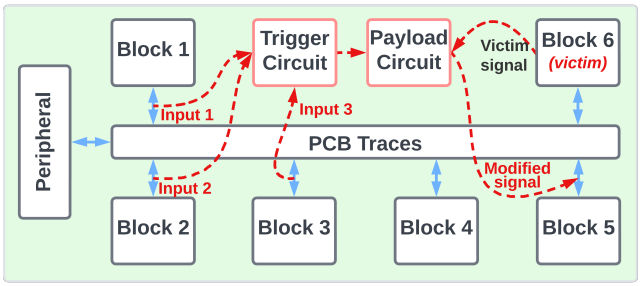


Figure 2.4: Sample diagram of a PCB level triggerable functional hardware Trojan.

The payload of an HT is the malicious activity that the Trojan performs when it is triggered. Trojans which are always on have only a payload circuitry, since there is no

need for a triggering mechanism. These will not have an evident, significant effect on the functionality of the PCB, since otherwise they would be obvious and easily detectable for post-fabrication error tests. Various trigger and payload designs are possible within this model.

Trigger Designs: Cleverly designed triggers for functional Trojans should be robust against unintended activations. Triggers designed by cautious adversaries can utilise combinations of PCB components such as resistors, capacitors, diodes and transistors. These combinations can be divided into three distinct groups - Resistor-Transistor Logic (RTL), Diode-Transistor Logic (DTL), and Transistor-Transistor Logic (TTL) [26]. Using these groups, the same Boolean logic function can be implemented in many ways. For instance, even the simple double input NAND function can be constructed in three circuits as shown in Figure 2.5.

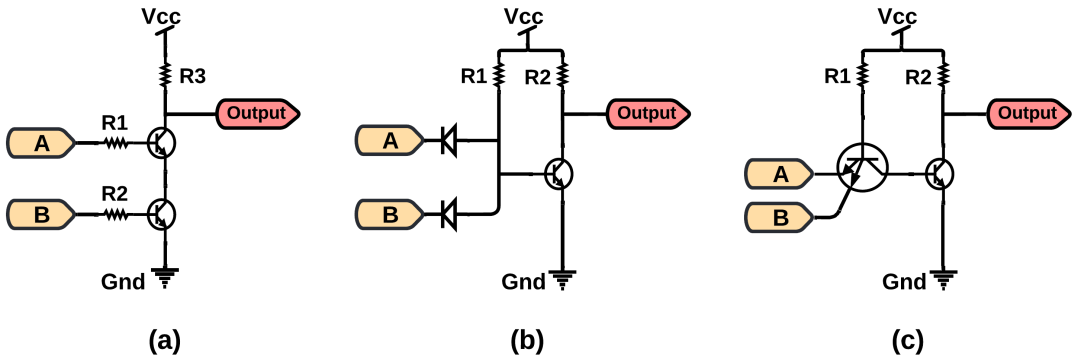


Figure 2.5: Schematic diagrams of a double-input logic NAND function with (a) resistor-transistor logic, (b) diode-transistor logic and (c) transistor-transistor logic.

The sample Trojan shown in Figure 2.6 illustrates the general concept of how it can modify a signal communicated between two microcontroller units (MCUs). Such Trojans can be introduced by untrusted design houses with little hindrance.

One example of the triggering circuit can be a triple input logic AND gate implemented in RTL, shown in Fig. 2.7. As such, the trigger function will only activate when all three triggering input signals are in Boolean logic high state. Evidently even this simple Trojan trigger design can be scaled up in the number of input signals, making it challenging to detect given the sheer magnitude of possible combinations [26].

Another example of a triggering circuit consisting of resistors, a capacitor, a diode

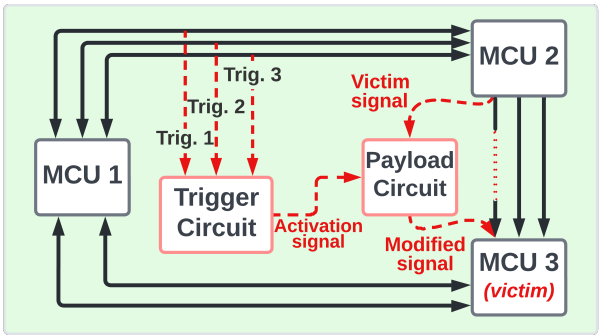


Figure 2.6: Triggerable Trojan implant modifying a signal going from microcontroller 2 to microcontroller 3 by rerouting the trace through its payload circuit.

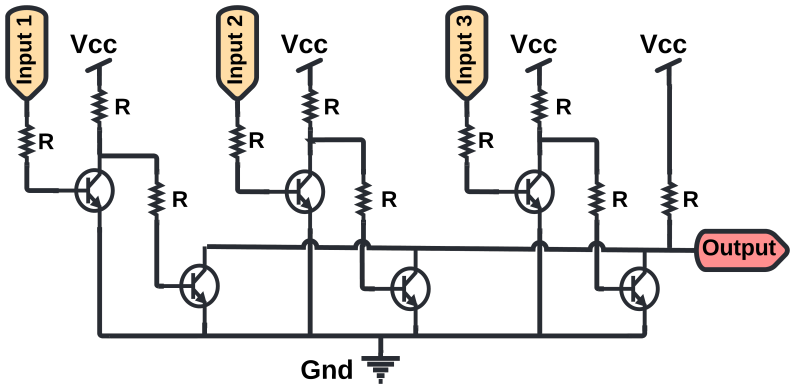


Figure 2.7: Schematic diagram of a sample multi-input trigger circuit, activating when all inputs are simultaneously in logic high state (3-input logic AND gate).

and an Op-Amp is shown in Figure 2.8 [27]. This is a more complex trigger design, which activates if the included capacitor is charged up to a certain voltage. The reference voltage is determined by the resistors in the Op-Amp feedback loop and the charging process takes place only when the triggering input signal is high. If the capacitor voltage does not reach the reference point before the input signal reverses to low, the diode blocks the capacitor from discharging. Alternatively the diode can be removed opening a discharge path for the capacitor, in which case the input signal will need to remain high for a minimum amount of time before the capacitor reaches its reference voltage.

Example Payload Designs: The same RTL, DTL and TTL methods can be utilised to implement the Trojan's payload functionality. Logic gate XOR can be used as the payload to invert the victim signal every time the trigger circuitry is activated. One such example can be seen in Figure 2.9, implemented in RTL. Alternatively a 2:1 Multiplexer can be

used for functional leakage or even a single transistor to cause a stuck-at-logic high or low (Figure 2.10).

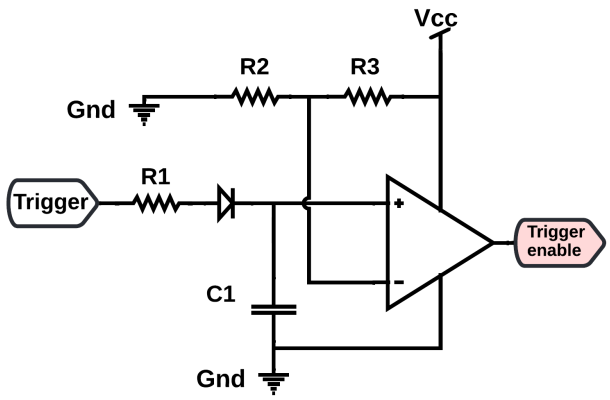


Figure 2.8: Schematic diagram of a trigger circuit using a capacitor and op-amp, which will activate only after several repetitions of the triggering condition.

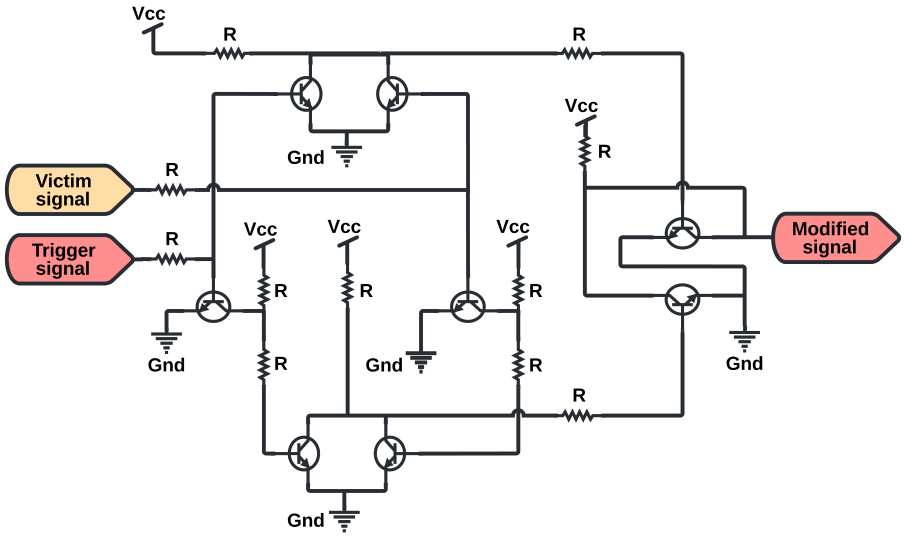


Figure 2.9: Schematic diagram of a sample payload function (2-input XOR) which inverts the targeted victim signal upon activation.

2.3.2 Parametric Trojan

When the PCB has fewer layers and hiding additional components is harder, harm can be done by changing the resistance, capacitance or inductance of the signal traces. For instance, the resistance can be increased by thinning the copper signal trace in an internal layer thus shortening the operational life of the copper trace due to heating.

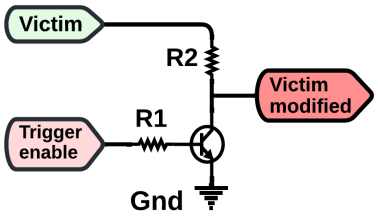
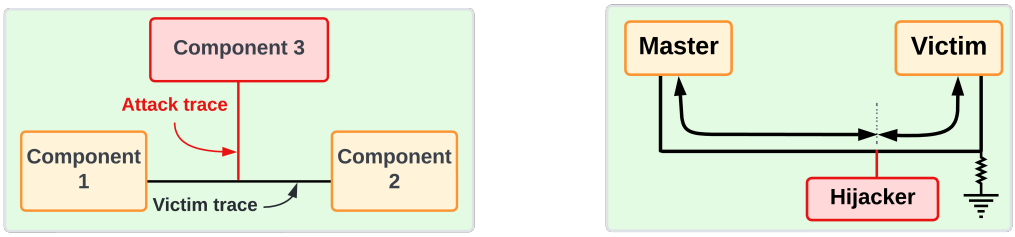


Figure 2.10: Schematic diagram of a single transistor payload circuit with a stuck-at logic low effect.

Several possible trace modification techniques were proposed in [6] and [10]. One of them was based on hijacking a signal state by changing drive strengths of different signals that are connected to each other.

For example, in case of some debugging techniques such as JTAG the circuit architecture requires linking the debugging pins to the pins of legitimate components on the board. In the schematic illustration of a design shown in Figure 2.11a, a trace from component 1 drives components 2 and 3. However the signal heading towards component 2 can also be controlled by component 3 by configuring the pin of component 3 as an output with a stronger drive than the output of component 1. The strength of a trace drive can be changed by altering its impedance, thickness and length, or integrating buffers (Fig. 2.11a) [6]. Another example is illustrated in Figure 2.11b. Although the driving power of the hijacker is lower than that of the master component, being situated closer to the victim element provides the hijacker a temporal window to generate a false signal before the correct signal from the master can reach the victim component.



(a) (b)

Figure 2.11: Diagrams of two parametric Trojan models: (a) Trojan Component 3 hijacking communication between Component 1 and Component 2, (b) Located closer to the victim, the hijacker generates a fake signal, reaching the victim faster than the genuine signal.

2.4 Taxonomy of PCB Trojans

Understanding different type of HTs and categorising them in a taxonomy is vital for developing different countermeasures against Trojans. With HTs becoming more complex, different taxonomies have been proposed for IC level Trojans since 2008 [28], [22]. The most comprehensive taxonomy for IC HTs was proposed by Karri et al. [29]. However existing taxonomies for IC-level Trojans are not suitable for PCB-level Trojans due to existing differences between IC-level and PCB-level Trojans (section 2.6), in particular differences in threat models, attack surfaces, and scale. For example, insertion of an HT after manufacturing is not possible at the IC-level, whereas it is a realistic threat for PCBs since malicious implants can be soldered on a PCB even after the fabrication stage [26].

The differences between PCB-level and IC-level Trojans determine not only the existing HT attack models on the supply chain, but also directions for developing countermeasures against PCB-level Trojans [20]. The first taxonomy for PCB-level HTs was proposed by Ghosh et al. [10] where board-level Trojans were categorised as malicious modifications or hidden components. More detailed and comprehensive taxonomy for PCB-level HTs was presented by Hoque et al. [26] where the taxonomy for IC-level Trojans was modified to accommodate the differences between IC-level and PCB-level HTs. Although later a taxonomy constructed specifically for PCB hardware Trojans was suggested by Harrison et al. [20], the taxonomy in Figure 2.12, based on a combination of the two, provides a more accurate and encompassing representation of PCB-level Trojans, showing classes of HTs, highlighted in yellow with dotted outlines, that are applicable only to PCBs.

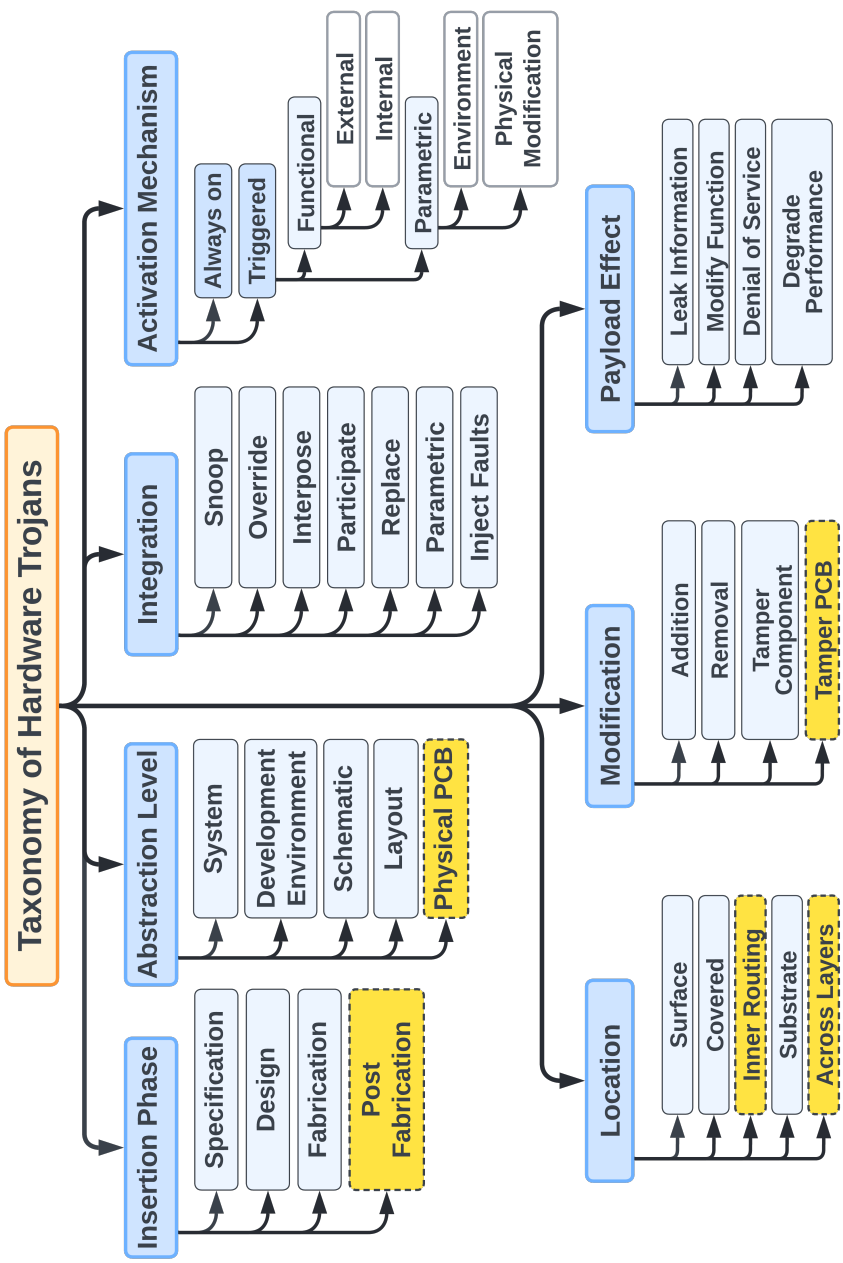


Figure 2.12: General taxonomy overview of HTs, binned in groups based on specific description mechanisms. The highlighted yellow boxes show the Trojan types applicable only to PCBs.

The first column in the taxonomy describes **insertion phases** in the PCB life cycle when malicious modifications are possible to introduce. A hardware Trojan can be implemented by modifying the PCB characteristics at *specification* and *design* stages, for example changing temperature specification to deteriorate the design reliability. Insertion of new elements or trace alteration are also feasible during the *fabrication* and *post fabrication* steps [8]. IC-level HTs can be introduced by adding extra gates to the design's netlist or directly changing the photolithography masks. After fabrication an adversary can also exploit electromagnetic radiation caused by unshielded wire connections to cause leakage of classified information or inject faults in the system [30].

The level of abstraction provides an insight on the extent of control and flexibility the adversary has. It includes different stages of the hardware IP development before fabrication.

- The *system level* defines the design's modules and their interconnections. At this level adversaries are limited by the modules' interfaces and interactions between them.
- At the *development environment* level an HT can be introduced into the modules by exploiting computer-aided design (CAD) tools and scripting languages.
- The *schematic* level represents a list of gates/components with interconnections between them. Here the adversary has enough information to apply a targeted attack through an implanted Trojan block and manipulate the signals on the legitimate interconnections.
- Access to *layout level* enables the adversary to fine-tune the HT in terms of the temporal delay and additional power consumption it introduces to the original device. In fact these are parameters, which can be regulated with very high precision. Further, apart from newly added HTs in the circuit, HTs can also be introduced through modifying the parameters of existing legitimate components in the circuit.
- Lastly, the *physical level* provides the adversary access to the exact placement, as well as physical dimensions of all components on the device. It would be possible to implant an HT in the unoccupied areas of the original device layout, without impacting its initial characteristics [30].

The location characteristic describes where in a PCB an HT can be implanted. It can be on the top or bottom of a PCB and be visible without disturbing other components.

Trojans could be located on the surface of a PCB, but be hidden by a quality control sticker, heat sink or another component. They can also be hidden between the layers of the PCB, in the form of embedded extra components or modifications to routing and cause, for example, a signal propagation degradation or cross-talk aggravation [10], [31].

On a PCB housing several chips, an HT can be implanted on chips' interfaces to modify communication.

HT Modification discusses the ways an HT is introduced into a PCB.

- A Trojan can be introduced by *addition* of an extra component on empty spaces or test points of an original design. Traces can be added for transferring HT signals or prompting crosstalk. Added test points can uncover sensitive signals that are hidden in inner layers. The Trojan described by Robertson et al. [2] is an example of an HT introduced by addition.
- A Trojan attack can be arranged by *removing* components in order to change or jeopardise system's functionality. For example, by removing one of the existing capacitors soldered to the output pins of a switch mode power supply the system voltage can be destabilised at the times of peak current demand.
- *Tampered component* group is about maliciously modified components in the supply chain, which are later unknowingly used by trusted entities down the production line. For example, if a certain microcontroller is tampered by an adversary and presented in the market as an original trusted component, a trusted PCB manufacturing facility can purchase that microcontroller and use in its production line, thus unintentionally introducing a compromised component to the otherwise trusted product.
- *Tampered PCB* is about modifying the original design or replacing a legitimate component. For example, the PCB substrate or routing layers can be modified by an attacker. An example of this type of HT attack is introduced by Robertson et al. [2], describing how the crosstalk phenomena can be exasperated by changing the routing path of a PCB trace. Similarly, [12] discusses how electromigration induced failures can be introduced by tampering the physical dimensions of a trace, e.g. thinning a copper wire. Another example is camouflaging an active HT component as a passive component, which replicates the task of a passive component with added malicious functionality. Likewise, legitimate ICs could be knowingly replaced by

Trojan counterparts, e.g. an IC carrying a small logic could be replaced with a similar looking, but more powerful Trojan microcontroller [12]. Such Trojans have the capability to add or change the existing system functionality.

Activation mechanism category describes the mechanisms by which an HT payload is activated.

- *Always active* Trojans continuously deliver their payload during the operation of the system. Examples of this type of HT include replacement of a passive component in a PCB by another passive component with skewed parameters or modification in the routing of an otherwise shielded trace, for example, so that information on the signal carrying trace can be leaked through electromagnetic radiation.
- *Triggered* category HTs are divided into functional and parametric types which are described in section 2.3. Triggerable HTs need to meet some predefined activation condition before they initiate delivering their payload. Anything from internal sequential counters or external data streams entering the system, to variation in the power supply voltage, and even environmental temperature can serve as an activation trigger.
  - *Internal triggers* can be activated when encountering specific electrical conditions or system states, such as interrupt signals and long periods of high-frequency switching [32].
  - *External trigger* activation can occur as a response to a signal or system state caused by external intervention. The trigger of the HT can be programmed to activate in case of a specific input sequence to the device from one of the legitimate input ports (e.g. signal sent from a neighbouring device). The activation command can also be delivered directly from the adversary through wireless communication channels, one of the physical input ports or even through active manipulation of the levels of supply voltage to the device. Another more rudimentary yet applicable example of an external trigger can be a system of mechanical relays activated manually with an external magnet [20].
  - *Environmental trigger* activation is due to conditions which are independent of the original function of a system. These could be time, temperature, aging or the presence of radiation. For instance, thinning PCB traces at the surface

layer from heating due to a long period of operation can be used to trigger early system failure [12].

– *Physical modification* can also serve as triggering mechanisms. One such example is a case where a metal trace has been reduced in width, below the permissible threshold given by the design specifications. This will result in overheating of the trace, which in turn can result in trace failure under heavy workload, thus triggering, for example, a denial-of-service Trojan.

Payload effects range from *modifying the functionality*, for example, by adding an extra component in the circuit to manipulate signals going from one IC to another, *degrading the performance* or deteriorating the reliability of the PCB by modifying its physical parameters, up to complete failure of hardware operations, i.e. *denial of service* [30].

The integration of a Trojan characterises how a Trojan affects signals and components in a PCB. The aim of a *snooping* HT is extracting information i.e. reading signals without modifying them [33]. A Trojan can also *override* signals to affect the operation of the system. Not all signals are subject to be overridden, only open-collector/open-drain signals or signals driven by medium impedance can be successfully overridden [34]. *Interposer* Trojan attacks break one signal into two making it possible to control the signal without creating a conflict [19]. Another possibility in this category are Trojans that can *participate* in communications with legitimate components and extract data [35] or, for example, gain control by imitating an administrator [32]. An HT can be inserted as a *replacement* for a legitimate element [36] or affect existing parameters such as trace impedance, electromagnetic properties, heat dissipation etc. [12]. The last type of HT under this category can *inject faults* by making other parts function outside their legitimate role [37].

2.4.1 PCB Taxonomy and Countermeasures Against Trojans

The taxonomy built in Figure 2.12 can help identify Trojans according to their characteristics. In particular, PCB Trojan countermeasures will depend on the modification induced by the Trojan. Side-channels will be differently affected by the addition, removal or tampering of components in a PCB. Addition of an active component

will cause increased power consumption, while tampering may not change it noticeably. Types of modifications will significantly affect both impedance and signal reflections. For optical countermeasures different algorithms may be required to fight against added, removed, and modified components. For example, obfuscation-based countermeasures could prevent adding a component on a PCB, but an attacker may replace a system's FLASH memory by an HT chip without breaking obfuscation [20].

While an HT's location categorisation may be less important for side-channel based countermeasures, they may play a key role for multi-modal imaging based detection. Different imaging methods are needed for detecting HTs located in different places on a PCB. For example, optical cameras cannot detect modifications to the inner-layer routing, but they can help effectively verify IC serial numbers. Trigger and payload types are important for functional testing and other countermeasures which are most efficient for already activated Trojans.

Integration characteristics are also important for determining countermeasures against Trojans since they highlight weaknesses in PCB security that allow Trojan attacks. For example, snooping would be possible through an insecure channel, a vulnerability which could be avoided during the hardware design phase. The HT integration characteristics can help detect the impact of a Trojan on the parameters of a PCB. Functional countermeasures also can be influenced by integration characteristics. For example, a Trojan that takes part in communication protocols between components could be detected by policy engines [20].

A comprehensive taxonomy is not only important for developing countermeasures against HTs, but it can also help to create benchmarks which would be helpful for evaluating effectiveness of countermeasure techniques for different classes of Trojans [24].

2.4.2 Hardware Trojan Benchmarks

Without publicly available test benchmarks it would be difficult to verify the effectiveness, strengths and weaknesses of different HT detection techniques. A benchmark for IC Trojans was developed by Salmani et al. [38] and published in the Trust-Hub in 2013 [39]. There are currently 96 HT-infected benchmark circuits in Trust-Hub, which has been widely recognised for evaluating the effectiveness of IC HT detection approaches. Trusted benchmarks with 24 types of PCB Trojans have been developed only recently [20] and

added into Trust-Hub in 2021 [38]. These are shown in Table 2.1 presenting PCB Trojan types with the respective number of affected samples, which partly cover some categories of the Trojan taxonomy presented in Figure 2.12. For instance, the table shows that 16 Trojans are inserted on the surface of a PCB, 4 are covered, 2 are located in inner routing. Under the payload effect category 9 HTs leak information to outside of a PCB, 16 modify the circuit functionality, 2 cause a denial of service when activated and 1 HT degrades the PCB performance.

Table 2.1: PCB Trojan Types with the Number of Samples in the Trust-Hub Benchmark

Trust-Hub Benchmark: 24 samples in total (without duplication)		
Location		Surface 16, Covered 4, Inner routing 2, Substrate 0
Modification		Adding 14, Removal 1, Tamper component 5, Tamper PCB 0
Activation Mechanism	Always on	7
	Triggered	Internal 10, External 6, Environment 1
Payload Effect		Leak information 9, Modify function 16, Denial of service 2, Degrade performance 1
Integration		Snoop 5, Override 9, Interpose 1, Participate 6, Replace 1, Parametric 3, Inject fault 0

This publicly available Trojan benchmark has only limited examples for each class of Trojan and, therefore, represents a small part of possible designs that adversaries could implement. In best case scenario the available data will be enough to train a machine learning model, which can detect similar HTs in real life with a reasonable accuracy. However, the model will probably suffer from lack of sufficient generalisation, i.e. the model will struggle to classify the types of HTs it has not encountered in the dataset during the supervised learning model training phase.

In addition, machine learning based Trojan detection methods become increasingly effective and require a large number of training data, whereas a few available examples in the Trojan benchmark suite may not allow to create a reliable trained model able to capture malicious modifications.

To address the above-mentioned shortcomings Hoque et al. [26] developed a tool for automatic PCB benchmarking. Different models of PCB level Trojan designs have been developed and a custom pool of Trojans of different complexity generated. Using a Trojan insertion tool these Trojans were automatically inserted into various PCBs to generate a large number of HT infected PCB designs i.e. Trojan benchmarks. Taking the netlist

representation of a sample PCB design as an input the tool applies KiCAD PCB design software [40] to generate the netlist containing a Trojan. The process can be repeated to generate as many Trojans as required. As an example, the tool generated 150 PCB Trojan benchmarks if provided with the netlist of five different triggers and three payloads.

Since this benchmark is based only on KiCAD.NET format netlists for generating PCB layouts with or without HTs, it is restricted in use in terms of, for example, assessment in the schematic level of abstraction for the exact same design. In addition, Trojans introduced in these benchmarks are made up of separate components such as resistors, and capacitors, to imitate the operation of the logic gate. This excludes a situation where adversaries can use off-the-shelf small package chips, such as widely available integrated circuits, to carry out more sophisticated attacks.

A more systematic framework for classifying and identifying more advanced board-level HT attacks and a comprehensive benchmarking method for such attacks is developed by Zhu et al. [41]. The authors have suggested practical benchmarking rules and a workflow of generating PCB attack benchmarks. Here, the attack scenarios have been divided into well defined cases whereby three stages of PCB lifecycle have been discussed with five general types of malicious alterations. Additionally, four PCB-specific design rules have been considered in order to accurately model the attack scenarios for the PCB level of abstraction.

2.5 Integrated Circuit Trojan Countermeasures

The problem of hardware Trojans appeared as an important research area for hardware security in the late 2000's, after a large number of cases were reported on HT attacks inside integrated circuits (ICs). These attacks occurred in all stages of the IC production cycle and the attack models were different depending on which stage of the design or manufacturing flow they occurred.

The first comprehensive analysis of hardware Trojan attacks, defence strategies and countermeasures was presented by Bhunia et al. [37]. The authors also suggested taxonomies for Trojan types and countermeasures against Trojans. In this section the research on countermeasures against IC Trojans is reviewed, categorised in four broad groups. Three are under conventional countermeasures: Trojan detection, Design for security, Runtime monitoring (Fig 2.13) and one is under Machine Learning approaches (section 2.5.2, page 31). The detection approaches, as the name implies, aim to detect HTs in a circuit. Design for security approaches aim to prevent or hinder inclusion of HTs at design phase. Research on runtime monitoring focuses on identifying HTs and effects on the IC from their activation [42].

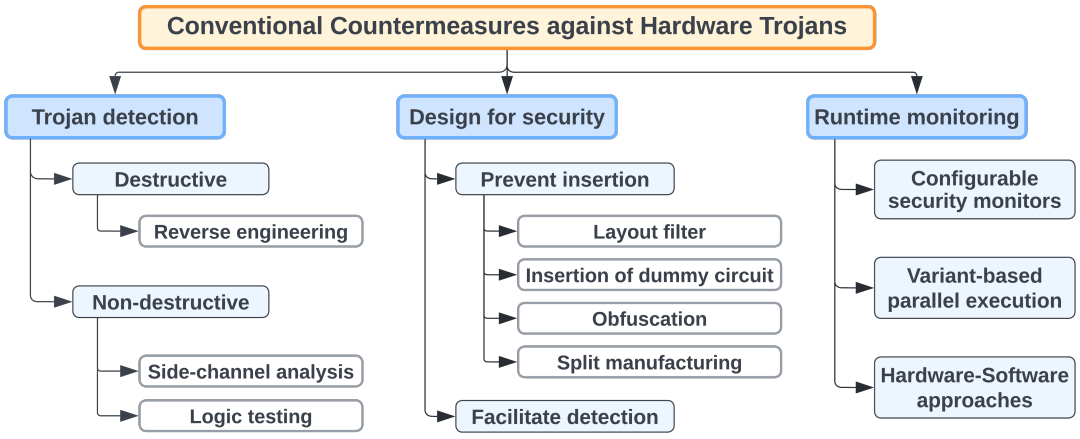


Figure 2.13: Countermeasures against hardware Trojans.

2.5.1 Conventional Countermeasures

2.5.1.1 Trojan Detection

HT detection is the most commonly applied countermeasure against Trojans. Detection methods authenticate the design and manufactured ICs without additional circuitry and can be applied both in the design stage and after the manufacturing stage (post-silicon) [25]. Post-silicon detection methods can be destructive or nondestructive.

Destructive. Destructive techniques normally use reverse engineering (RE) to visually verify an IC against its golden model. Although these methods give the highest assurance against malicious implants, they are too costly, could take a long time to conduct and destroy the IC during the process, while providing useful information only for a small batch of IC samples. Therefore destructive methods are not practical for HT detection at large scale [37].

Non-destructive. Non-destructive methods try to verify fabricated ICs received from an untrusted foundry via side-channel analysis or logic testing.

Side-channel based approaches: There has been a dramatic increase in the number of publications since the original paper by Agrawal et al. in 2007 [43], where the first side-channel based approach was proposed for detecting the presence of HT circuitry in an IC. Side-channel analysis methods are based on measuring circuit parameters including supply or transient current and path delay, and detecting any changes which could occur due to malicious modifications. These methods are particularly effective for detecting HTs with larger footprint on the particular side-channel and developing test vectors, but can be less effective for detecting HTs with a smaller effect on the side-channel. In addition side-channel analysis can be sensitive to noise and requires a golden model as a comparison baseline.

In the first side-channel based approach [43] transient current parameters were used for effectively detecting both small and large HTs. Aarestad et al. [44] proposed an HT detection method based on the analysis of steady-state current measurements in an IC. The novelty is that this analysis is carried out simultaneously from multiple places on the

surface of the chip. Experimental results proved the effectiveness of the multiple supply port technique for detecting HTs as small as those consuming 8 μA current consumption. Another side-channel based approach was presented by Rad et.al [45] where the detection of HTs was based on sensitivity analysis of power supply transient signal under different inauspicious conditions. This study was followed by a new HT detection approach for gate-level characterization that applied thermal conditioning to determine the scaling factors of all the gates using linear programming [46]. A new gate-level characterization based investigation was carried out by Koushanfar et al. [47] where HTs were effectively detected with low process overhead. In another research [48] a scalable method for HT detection was proposed again using gate-level characterization, this time based on segmentation of a large circuit into smaller sub-circuits. Leakage power profiles created for each sub-circuit increased the effectiveness of an HT detection since any modification caused by an HT would affect the total power leakage.

A new technique for IC authentication, Physical Unclonable Functions (PUF), is developed to extract unique signatures from an IC based on physical characteristics such as signal propagation delay [49]. Many PUF designs, most being circuit-delay-based, have been used to demonstrate the possibility of use in IC authentication. A delay-based PUF for HT detection was proposed by Li and Lach [50], where a sweeping-clock-delay measurement technique was applied for selected register-to-register path delay measurements. An HT was detected when one of path delays was extended above the threshold defined by the process variation's level.

An interesting method for effectively detecting small HTs was proposed by Xiao et al. [51] where a clock sweeping technique was combined with transition and path delay patterns for generating delay signatures. By analysing the transmission power differences between amplitude signals and the frequency a method was developed by Jin and Makris [52] for detecting HTs which aim to leak information from wireless cryptographic ICs. A multiple-parameter side-channel analysis using dynamic current and maximum operating frequency was also carried out by Narasimhan et al. [53] where the effectiveness of HT detection was improved by using a sensitivity technique.

Logic testing. Logic or functional testing methodologies develop test patterns for activating HTs. They evaluate the behavior of internal nodes and initial outputs of the

circuit based on a set of input vectors. If the deviation in the evaluated parameter is suspicious, the design is considered Trojan inserted. Logic tests can therefore be performed at any stage of IC design and were developed to overcome the shortcomings of side-channel analysis methods.

Shortly after the first side-channel based research [43] the first research was carried out on developing an HT detection approach based on logic testing [54]. The first research was followed by different logic testing based methods. An approach for finding isolated sections and weakly correlated signals in the netlist for identifying HT triggers is suggested by Cakır and Malik [55]. Oya et al. [56] carried out comparison of Trojans to determine frequently used architectural patterns in HT design. A score-based classification is then implemented to detect HTs in trusted netlists. Also, a golden model can be generated to perform a formal verification in the case when a trusted specification is available. However HT detection becomes very challenging if the trusted specification is not available.

Research papers [57] and [58] test generation strategies are presented for optimization of the number of test vectors needed for an HT activation. A new HT detection technique is proposed by Zhang et al. [59] which targets the design stage and is based on the detection of HT trigger inputs. This approach involves a tracer and a checker, where the tracer identifies the trigger signals with inactivated entries and the checker analyzes the signals to determine ones that are associated with HTs.

Logic testing and side-channel analysis were the first countermeasures against HTs. Logic testing is only effective for an active HT, an HT with a known design and for identifying suspicious nodes in the circuit. Side-channel analysis is capable of detecting even inactive HTs, but in this case a golden model is required. This approach also has problems in dealing with process variation masking effects which makes the detection of small HTs challenging. Both side-channel analysis and logic testing methods have advantages and disadvantages, but a combined implementation of the two approaches can provide higher effectiveness for detecting different HTs implanted in more sophisticated ICs [53].

2.5.1.2 Design for Security

In addition to detection methods, other design-level strategies have been developed to prevent a Trojan insertion at hardware design stage or facilitate test time or run-time

HT detection. These methods can be collectively named design for security. The first publications for HT prevention based on design for security appeared in 2012 [58], [60].

Prevent insertion. Preventative methods can in turn be classified into four broad classes: layout-filler, insertion of dummy circuit, obfuscation, and split manufacturing.

Layout-filler methods are applied to fill the empty gaps in an IC with filler cells thus preventing the inclusion of malicious components. A layout-filler technique for HT prevention in the layout phase was introduced by Xiao et al. [61] where functional filler cells were used to fill the unused spaces in an IC. If the presence of HTs changed any of the filler cells, a self-testing procedure generated a digital signature. This method detected different instances of an HT insertion without requiring a golden model and showed negligibly small power, area, and time overhead. Another layout-filler technique for field-programmable gate arrays (FPGA) was suggested by Khaleghi et al. [62]. The proposed approach significantly reduced the possibility of a Trojan attack since there was no free configurable resource for HT inclusion, without performance and power overhead.

An HT can also be inserted into an IC through rare nodes with low transition probability. To prevent the inclusion of such Trojans, *dummy circuit insertion* approaches were developed, where dummy scan flip-flops were used to reduce the rare nodes and prevent the insertion of an HT [63]. Dummy flip-flops and analysis of the transition generation time for facilitation of an HT detection also was carried out [58].

The obfuscation method changes the transition mode and makes it difficult to understand the IC design. Obfuscation is considered best-possible if the design after obfuscation leaks no more information than other designs for the same action. This approach makes the insertion of a meaningful malicious component into an IC more difficult.

A theoretical investigation of obfuscation was first carried out by Barak et al. [64] where a black-box obfuscation was proposed, which required that the obfuscated IC performed the same function as the original IC, and leaked no information other than its black-box (input– output) functionality. This definition of obfuscation, however, is not always realistic to achieve. Later a new definition of a best possible obfuscation was suggested by Goldwasser et al. [65] where the second requirement was lightened.

The obfuscated IC was required to leak no more information than any IC with the same functionality. A new PUF construction method for extracting the unique power-up state for each IC was proposed by Xue et al. [66]. Through the proposed obfuscation method, which uses PUF based unique key sequences, it is possible to control the IC's operation modes and functionalities, as well as remotely disable the chip when an HT is detected. The functional obfuscation prevents the attacker from recognising both the real functionalities of the ICs and the real rare events in the internal nodes, making it hard for the attacker to insert more sophisticated HTs.

In general obfuscation approaches can be classified into combinatorial and sequential logic obfuscation. The first is to obfuscate IC designs by randomly inserting additional key-gates [67], [68]. This does not necessarily guarantee that wrong keys will corrupt the outputs. To circumvent this problem Rajendran et al. [69] proposed a fault analysis-based encryption approach which was able to ensure that wrong keys would corrupt the outputs and thus maximise the uncertainty for an adversary. The second logic obfuscation technique is sequential, when new states and transformations are added to the finite state machine [70], [71]. Authors in [72] showed that the best possible obfuscation for a sequential circuit can be achieved by pursuing consecutive four steps: re-timing, re-synthesis, sweep, and conditional stuttering. If the correct key is applied, the obfuscated IC design will function correctly.

Split manufacturing concept is in splitting the circuit into two different parts with one part including the transistors and some of routing wires, and the second part containing the rest of routing wires. The two parts are manufactured in different foundries, one in an untrusted third-party foundry and the other part is fabricated by a trusted foundry [73]. This method hides the main functionality of the IC design, thus hindering the attacker from copying the design or modifying it by inserting HTs [74]. A few approaches have been proposed in recent years to prove the feasibility of split manufacturing [75], [76], [77].

Facilitate detection. HT detection facilitation approaches are applied to assist in detection of the location, category and triggers of HTs. This is achieved by combining functional testing, side-channel analysis and runtime monitoring. Various such approaches have been proposed in [78], [79] and [80].

2.5.1.3 Runtime Monitoring

The techniques developed for HT detection and prevention have been complemented by runtime monitoring methods, which try to ensure the circuit stops functioning and security mechanisms are triggered in the case of any suspicious action in the circuit.

Runtime monitoring has been comprehensively discussed and classified into three categories: configurable security monitors, variant-based parallel execution, and hardware-software approach [23]. Within *configurable security monitor* techniques the authors of research [81] proposed a runtime monitoring approach for detecting HTs, which could leak classified information. In 2009 a first runtime detection method based on *variant based parallel execution* was developed by McIntyre et al. [82]. This approach is based on simultaneous execution of functionally equivalent variants obtained by different algorithm variations and comparison of the results. The process dynamically achieves the detection of an HT with a high level of confidence. This work was followed by the first research based on *hardware-software approach* [83]. Runtime monitoring can also be carried out by exploiting analog sensors, which can detect deviations in power or thermal profiles caused by Trojan activation [84]. It is now accepted that a complete solution to combat HT attacks can be reached by combining HT detection, prevention and runtime monitoring [23].

Research on HT countermeasures has been reviewed in several articles. These include comprehensive reviews, where the complex HT threat models and feasible countermeasures in certain areas of HT attacks are illustrated [23], [85]. Reviews for an HT detection in intellectual property (IP) cores and fabricated stage, and a survey on an HT threat and defense have also been carried out [86], [87]. Later, different HT detection techniques, as well as the complexity and limitations of those techniques were analysed [88], [42].

2.5.2 Machine Learning Approaches

Machine learning (ML) techniques have the potential to contribute to the development of countermeasures for various HT attacks [89], [90]. The first ML classification model was introduced by Jin et al. [91] in 2012, when an artificial neural network classifier was developed for detecting HTs. Since then, IC HT detection ML methods have been

widely implemented, improving the detection capabilities in a number of aspects including reverse engineering [92], side-channel analysis [93], [94], real-time detection and for gate-level netlist detection [95]. A comprehensive survey of the latest research and developments of ML-based approaches for HT detection and prevention on an IC level is carried out by Huang et al. [96], who have demonstrated that the number of research publications and achievements in IC HT detection using ML techniques have been growing rapidly since 2014. Several other surveys including [92], [90] and [42] discussed and analyzed ML techniques applied for HT detection, classification and prevention.

The machine learning HT countermeasure approaches can be categorised in five broad sections: Reverse Engineering Improvement, Real-Time Detection, Golden Model Free, Gate-Level Netlists Detection and Classification (Fig. 2.14). Albeit, at this level of detail there may be some overlap between the named groups.

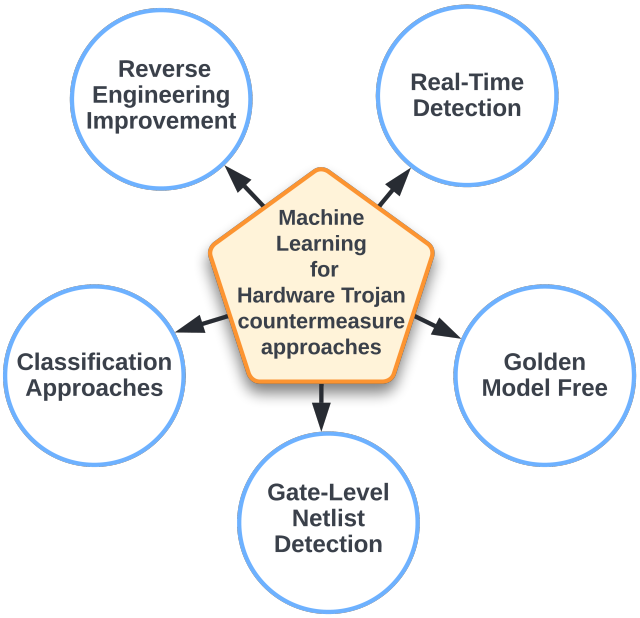


Figure 2.14: Machine Learning countermeasures against hardware Trojans.

Reverse engineering improvement. Reverse Engineering (RE) techniques have been introduced for inspecting the production of ICs. They consist of several steps which make RE time consuming, costly, and prone to errors. Reverse engineering improvement (REI)

aims to improve the reverse engineering during the imaging phase and the execution time and minimise the errors during RE. This approach was first introduced in 2014 [97], when One-Class Support Vector Machine classifier was applied based on RE images received from the imaging phase. By eliminating two RE steps the authors significantly reduced the overhead without a need for golden models.

The same authors later proposed K-means clustering based REI approach for identifying a Trojan free IC and three types of malicious modifications: one component duplicated, one component removed, and the width of one gate doubled. This approach demonstrated high accuracy for HT detection without requiring golden models, but was too sensitive to noise [98]. Another novel reverse engineering improvement approach for HT classification and detection during fabrication stage was proposed by Abdurrahman et al. [99]. Three types of malicious modifications were introduced to reflect Trojan insertion, deletion and parametric modification. The changes were accurately detected and classified, however the performance deteriorated with increased noise level.

Real-time detection. These techniques are able to prevent an HT attack by detecting it, disabling or bypassing the damaged IC and securing its normal operation. Real-time anomaly detection was first proposed in 2016 by Kulkarni et al. [100]. Three different attacks, (traffic diversion, route looping, and core address spoofing attacks) were detected by using ML techniques. A novel real-time detection approach was also suggested by Kulkarni et al. [101], [102]. The authors compared four ML algorithms and proposed an online learning framework for detecting HT attacks at the design or fabrication stage. A novel neural network design and a feature extractor training was applied by Faezi et al [103] for run-time HT detection without requiring a golden model.

Golden model free. Golden models, which are essential in most side-channel based approaches for creating a reference model for expected side-channel values, are time-consuming to create and not always available [103]. Golden model free methodologies try to detect and classify Trojan affected ICs without requiring a golden model. This approach was first suggested in 2016 by Jap et al. [104] and was based on One-Class Support Vector Classifier and side-channel leakage measures. The results from simulation showed that the method could differentiate HT free and infected gates without requiring a

golden model and held a high HT detection accuracy and low complexity even under high level of noise. Another HT detection novel method was based on two-class classification [105]. With high accuracy and recall rate, it could detect known and various unknown HTs at the fabrication stage without the need of a golden IC.

A golden model free method was also suggested by Lodhi et al. [106]. In this research the power profile of a microcontroller's instruction set was extracted and used to train four ML algorithms. This then was applied for runtime HT detection. In another work [107] an unsupervised ML clustering algorithm was applied to separate genuine logic gates from HT circuits, based on two characteristics in gate-level netlists: controllability and observability. The proposed technique was able to fully recover the inserted HT and isolate its trigger and payload circuits without using a golden model or HT activation test patterns.

Recently an ML based chip-free HT detection method was implemented for detection of command-activated HTs [108]. In the proposed algorithms the authors apply three novelty and outlier detection methods (One-Class Support Vector Machine (SVM), Local Outlier Factor (LOF), and Isolation Forest) as distinguishers and tested the effect of each distinguisher in five different scenarios. The approach could first determine the parsing path of the hardware then find all the executable commands. By comparing the executable commands with the official command-list the method could find unregistered executable commands and an HT activated by these commands.

Gate-level netlist detection. ML based gate-level netlist detection methods analyse gate-level netlists features to detect anomalies in the behaviour of infected ICs. This method was first proposed in 2016 by Hasegawa et al. [109]. This novel static method applied an SVM algorithm and used gate-level netlist features at the design phase to identify most of the HT infected nets. Hasegawa et al [110] then applied another ML approach based on a random forest algorithm for HT feature extraction. In 2017 Inoue et al. [111] again applied an SVM classifier for gate-level netlist features and could detect only a subset of HTs, whereas the ML model suggested by Hasegawa et al [112], trained on a set of HT affected and HT free set and a new set of boundary nets, could identify most of the HT nets.

Classification approaches. These approaches classify infected and uninfected ICs based on an HT's different features such as dominant attributes and side-channel measurements. In the first publication [91] a general architecture for HT detection was presented based on on-chip measurement and one-class classifiers. The proposed method classified the trusted and untrusted regions and executed a post-deployment trust evaluation. An SVM algorithm and frequency domain features were used for a new HT detection technique [113], where the power consumption waveform data was converted from time into frequency domain through discrete Fourier transform.

In another research [114], the proposed hardware Trojan detection method is based on an extreme learning machine (ELM) neural network. Dynamic power consumption features were used to train the classifier. The experimental verification showed that this method could identify HT infected ICs with high accuracy. Lodhi et al. [115] suggested a self-learning framework based on timing features such as the delay and carried out a comparison between three ML methods, decision tree (DT), Bayesian Classifier and K-Nearest Neighbours (KNN), applied during the testing phase. The results showed that amongst the three algorithms the DT model detected HTs with the highest accuracy. Another set of ML algorithms, DT, KNN and SVM, for HT classification were applied by Noor et al [116], and it was shown that the DT and KNN learning models could correctly predict about 83% of the test data.

2.6 Printed Circuit Board Trojan Countermeasures

While considerable research has been carried out on the classification and detection of HTs inside integrated circuits, board level Trojans have been much less investigated. Although many HT detection methods developed for ICs can also be adapted to be applied for PCB HT detection, the differences between IC and PCB level Trojans require new countermeasure approaches to combat HTs at the PCB level.

Manufacturing test too can be useful for detecting various simple attacks and defects including errors made in component selection, temperature activated malicious inclusions, material or lamination defects, and out-of-tolerance functional changes. For example, a significant unauthorised change in a passive component's (e.g. resistor) rated value can have an effect of a ticking time-bomb in the circuit, by triggering a slow chain reaction of circuit degradation [12]. Although manufacturing tests may not be effective against well hidden (latent) defects and rare activation HTs, they have certain advantages that are worth discussing.

2.6.1 Manufacturing Test

In the past years several PCB assurance approaches have been developed [117], [118] which can be categorised as *bare-board testing, JTAG boundary scan, functional testing, in-circuit testing*, and *automated visual inspection* [5].

Bare-board testing checks traces and the board substrate for continuity and impurities that could affect electrical or thermal characteristics in the manufactured PCB [119]. Bare-board tests are conducted before a PCB is populated and cannot detect Trojan inclusions and other malicious modifications.

JTAG boundary scan, functional testing and in-circuit testing are assembly tests that check that PCB components are correctly located and soldered and operate as expected [120]. JTAG boundary scan and functional testing are not highly effective for HT detection because comprehensive functional testing is not viable and HTs designed to activate under rare conditions are unlikely to be triggered by test patterns [23], [121].

In-circuit testing (ICT) uses electrical probes to carry out an electrical testing and

can detect faulty solder joints, missing parts, open connections, etc. However ICT is not feasible for dense, multi-layered PCBs and can be costly due to the need for developing several test setups for different boards [5]. Besides, ICT does not test the functionality of the PCB and cannot be effective for detecting connection faults and HTs which do not affect the tested nodes [122].

Automated visual inspection: Especially important are automated quality assurance (AQA) methods which, being non-destructive, could detect a wide range of HTs on a PCB with minimal human involvement [5]. One of these methods is automated visual inspection for PCB verification and authentication (Chapter 5). This method is about testing on assembly lines and replacing human oversight, thus reducing the risk of unauthorised modifications during assembly. This approach can be used in different stages of the PCB life cycle and can be categorised based on the imaging modality and image analysis techniques used for PCB AQA [122]. Several automated visual inspection methodologies have been suggested including canonical image processing methods for detecting trace and via level defects [123], convolutional neural network for detecting similar defects [124], automated detection for component placement by directly comparing golden and test PCBs [125], [126] and text detection on the PCB for verification purposes [127], [128]. The most significant advantage of automated visual inspection is its ability to detect a wide range of malicious implants, especially if the golden models are available. However, this testing approach is not applicable for functional verification.

Thus, the existing PCB assurance methods have advantages and limitations. Although manufacturing tests may be able to detect Trojans which have payloads that remind defects, in general they are not equipped to detect sophisticated and stealthy modifications introduced by adversaries. With hardware attacks becoming increasingly sophisticated there is an urgent need for new detection methods. Automated visual inspection is very promising, because, apart from the advantages listed above, it can be applied to detect defects and hardware attacks in many stages of the manufacturing process.

In what follows the board level Trojan countermeasures are discussed following a similar division as for IC Trojans (Fig. 2.13) and as suggested for PCB HTs by Bhunia et al. [23]: *detection, design for security* and *runtime monitoring*.

2.6.2 Printed Circuit Board Trojan Detection

Due to the differences between PCB and IC Trojans, the supply chain and post-manufacturing attacks also can be different, requiring new approaches in board-level HT countermeasures [20]. Various examples of board level Trojans have already been presented [10] and new attack vectors considered and detection methods proposed [12].

HT attacks on PCB level were first explored in [10], where the authors demonstrated a high frequency PCB design where an adversary altered both the inter-trace spacing and the individual trace dimensions in internal board layers. This type of HT attack could also be detected by an X-ray inspection of the board after production. Existing research on PCB security is mostly concerned with detection of counterfeit PCBs [7], [129] and in-field tampering [8].

It is worth mentioning that some methods, such as short delay defect, anti-reverse engineering and transition error testing, developed for IC HT detection can also be used at a higher level of abstraction [130], [131]. Analogous to ICs, PCBs also have side-channel fingerprints, which can be disrupted by a malevolent modification of the substrate or routing. Several static or runtime techniques have been proposed for detecting and measuring these modifications on PCBs. Without experimental validation Ghosh et al. [10] proposed multi-parameter side channel analysis for verifying security-sensitive nets against a simulated performance.

Previously unexplored board level HT attacks and potential solutions have been discussed by McGuire et al. [12]. Here examples of PCB modifications and an in-depth analysis of a malicious attack, that cannot be detected via existing methodologies, have been presented. It was suggested to use the resonant frequency of PCB traces and observe the change in the thickness of traces introduced by a malicious attack. Hamlet et al. [132] considered the possibility of using different PCB routing patterns and passive components to generate a PUF response. They also indicated several ways of collecting and processing the PUF response both in static and runtime environments.

2.6.3 Design for Security

Design for security approaches aim to prevent Trojan insertion by removing HT attack surfaces or making it impossible for attackers to understand a design. One of these methods is anti-reverse engineering which tries to protect the intellectual property (IP) of PCB designers and can act as a preventative measure against PCB HTs.

Quadir et al. [133] presented a survey of anti-reverse engineering techniques and suggested that the possibility of PCB reverse engineering, probing and tampering attacks can be reduced by simply adding routing layers, routing sensitive signals in internal layers and hiding the inter-chip connections on the PCB. Similar suggestion that security-sensitive signals routed in inner PCB layers could prevent a tampering and probing attack was made by McGuire et al. [12].

More advanced obfuscation was proposed by Ghosh et al. [10]. They suggested to add extra traces, vias, and components that could send dummy signals and mislead attackers from recognising the obfuscated ICs. Meanwhile, a secured JTAG protocol would block adversaries from illegally accessing a PCB's internal design information by using the test infrastructure.

A novel obfuscation approach was developed in articles [134] and [135], where the authors used a permutation network to obscure inter-component connections. The aim was to protect hardware against cloning, reverse engineering and unauthorised functioning. The approach was proven to be secure against brute-force attacks, but not against attacks that were guided by clues such as dedicated IC pin functions [136].

2.6.4 Runtime Monitoring

Similar to the IC Trojans case, runtime monitoring ensures the security of PCBs in the field by watching side channels and detecting divergences from expected behavior. This can be done, for example, by monitoring deviations in passive characteristics of traces which change when PCBs undergo probing, tapping, or alteration during an HT attack. In runtime monitoring Paley et al. [8] suggested to measure the deviation in resistance caused by an addition of a drop of solder and monitor the deviation in real time to find out if the PCB has been tampered. Similar approach based on conducting online capacitance

measurement in a circuit was suggested by Nishizawa et al. [137].

PCB ring oscillator PUFs were used by Guo et al. [138] to detect deviations in impedance caused by tampering or probing. The main idea was that the presence of a malicious inclusion would increase the trace impedance, thus increasing the frequency of the ring oscillator PUFs. Tampering would be detected if the frequency of the ring oscillator lay outside the expected margins.

To prevent counterfeiting PCBs similar ring oscillator PUF-based concurrent IC and PCB authentication was proposed by Zhang et al. [139] and Wang et al. [140]. The authors used the fact that the impedance of the PCB's power network shows an increase at the resonant frequency. First, a ring oscillator array was inserted into an IC. Then, the counterfeit PCB was detected by checking if the period of the ring oscillator matches that of the legitimate PCBs in the resonant and non-resonant states. The proposed authorization approach was implemented on legitimate and counterfeit field programmable gate array (FPGA) development boards, and over 90% counterfeit PCBs were detected by a two-class SVM classifier [139].

A novel runtime PCB HT detection method based on power analysis is demonstrated in [141]. Power consumption of individual legitimate components on a PCB and the total power consumed by the PCB were measured. These two measurements were compared and discrepancies larger than a certain tolerance were considered as evidence of HT activity. This is further discussed in Chapter 3.

2.7 Research Objectives

Literature review shows that despite the considerable research carried out on HT detection and prevention on ICs, the same problem lies largely unaddressed on PCBs. In January 2019, when the work on this PhD research commenced, even a quick search for research articles on the topic of HTs implanted on PCBs would reveal that there was a big gap in research on this topic. Despite the alarming reports of HT attacks in the course of or succeeding manufacturing of PCBs [2], there were only a handful of papers, most of which only discussed the importance of the problem, provided taxonomy of HT

varieties and suggested possible research directions [121], [123], [142]. Given the relative ease of access to PCB circuitry for the adversaries, compared to that of ICs, the hardware security research focus in this dissertation has been narrowed down to HT detection on PCBs.

The following research objectives are set to achieve:

- **Objective 1:** To detect a hardware Trojan on the printed circuit board in real-time, through policing of power consumption side-channel characteristic on the PCB, affected by the presence of an active hardware Trojan component.

  The first research objective is met in Chapter 3. However this leaves an open ended question: *What happens if the adversary designs the HT to operate on power not directly drawn from the PDN, but rather through one of the ICs on board, legally present there? What if the HT is linked to an I/O port, latched to the IC for power like a parasite?* The short answer is, the HT will be invisible to the proposed methodology remaining undetected. The second objective is set to solve this problem.

- **Objective 2:** To develop a novel approach using machine learning for monitoring power consumption of all the blocks present on the PCB. Instead of a more basic approach of putting upper and lower thresholds on the possible power consumption of every IC, here the methodology is taking advantage of the existing correlations between power consumption behavioral patterns of all ICs on the board as a whole.

  The second research objective is met in Chapter 4. Thus, Chapter 3 and Chapter 4 together cover the whole circuitry of the PCB, leaving no blind spots for an adversary to install an active HT undetected.

- **Objective 3:** To propose a novel visual inspection pipeline for detecting malicious inclusions, regardless of their power consumption status, i.e. both active and passive components, as well as ultra-low power components are detected. Use optical digital images as the imaging modality of choice.

  The third research objective is met in Chapter 5 by establishing a low-cost and fast PCB automated visual inspection technique for detecting hardware Trojans.

2.8 Concluding Remarks

In this chapter, the definition of an HT attack, vulnerabilities of the existing PCB supply chain and possible board-level security attacks have been discussed. Depending on which physical characteristics of a PCB are tampered, HTs are classified into functional and parametric and examples of each type of Trojans have been discussed. A comprehensive taxonomy has been developed. The relevance of the taxonomy for developing HT detection and prevention techniques has been explained. A detailed taxonomy is also important for creating benchmarks crucial for evaluating effectiveness of countermeasure techniques for different classes of Trojans. Finally, existing publicly available HT benchmarks, which can be used for PCB HT countermeasures evaluation, have also been discussed.

An extensive overview of existing research on IC and PCB HT detection techniques has also been carried out. The research on countermeasures against Trojans is categorised in three broad groups: HT Detection, Design for security and Runtime monitoring and detailed review for each group of research was presented. Since machine learning techniques have big potential as countermeasures for addressing HT attacks, advances made in a number of directions (e.g. reverse engineering, side-channel analysis, real-time detection, gate-level netlist detection) using machine learning were also highlighted.

The last section of this chapter summarises research objectives of the dissertation which have been addressed in Chapter 3, Chapter 4 and Chapter 5, respectively.

Chapter 3

Hardware Trojan Detection on a PCB Through Differential Power Monitoring

3.1 Introduction

In this chapter, a power analysis method is proposed for detecting HT components implanted on PCBs. An experimental setup, using a hardware prototype, is built and tested for verification of the methodology. The results confirm the ability to detect alien components on a PCB and provide directions for further research. The performance degradation of the original PCB due to the implementation of the proposed approach is negligible. The area overhead of the proposed method is small, related to the original PCB design, and consists of Sub Power Monitors of individual ICs on the PCB and Main Power Monitor for the overall power measurement of the PCB. To the best of our knowledge this research is the first work to develop a PCB HT detection methodology using power analysis.

The chapter is organised as follows: the description of the attacker model in section 3.2 is followed by the proposed approach in section 3.3. Then section 3.4 is devoted to the proposed methodology. Further, section 3.5 describes the experimental setup with the results discussed in section 3.6. Finally, the chapter is concluded in section 3.7.

3.2 Attacker Model

An HT implanted on the PCB can have different power sources, including: (a) a built in energy harvester/battery, (b) the mains supply, (c) the power distribution network (PDN) of the PCB and (d) an I/O pin of a legitimate chip. In case (a), a visual inspection

of the board can lead to detection of the HT, since an energy harvester or a battery are typically large in size and hard to conceal. In order to connect to the mains power supply, the HT should have external vias which are visible, hence case (b) can also be detected through image processing and comparison with a golden model [143]. Of these cases, (c) and (d) are the stealthiest, since the HT can be fully operational, while all the modifications to the original design can be hidden in the internal PCB layers. In these cases an HT can escape detection through visual inspection, and research for alternative approaches is necessary.

In this attacker model, it is assumed that the adversary can use any HT irrespective of its payload and trigger, as long as the malicious device consumes additional power. It is also assumed that the power to the PCB power distribution network goes only through the dedicated Main Power Monitor (MPM). Further, the IC and PCB design houses, as well as the firmware and intellectual properties used to design the ICs are trusted. The threat rises from outsourcing PCB production to overseas facilities, as well as from the possibility of interception while the device is in transit from the manufacturer to the consumer.

It is possible for the adversary to swap a trusted IC on the PCB with an HT infected and counterfeit (recycled, remarked) ICs, but this attack case is out of the scope of this work. However, as mentioned in the introduction, there is a considerable amount of existing research aimed at design, detection and prevention of IC HTs and counterfeits, which can be utilised to tackle the threat of swapping ICs.

3.3 Our Approach

In this chapter, a Differential Power Monitoring (DPM) approach is proposed to detect hardware Trojans on the PCB powered from the PDN. To the best of our knowledge, this is the first work on PCB HT detection using power monitoring. The approach is applied during in-field operation, as a run-time monitoring technique. Additionally, previous works can be used as complementary techniques (e.g. encrypting-decrypting communication between legitimate ICs [142]). Hardware Trojans are detected by measuring dynamic power due to HT circuit switching and static power due to HT leakage current. Continuous

measurement of power consumption provides information on the PCB's internal activities and HT activation. It is assumed that the PCBs are not defective, therefore the source of any diversion from the expected power consumption patterns is assumed an HT.

The main characteristics of the proposed approach are:

1. **Independent of the type of Trojan:** Any HT that consumes power higher than the pre-programmed detection threshold will be detected.
2. **In-field, online monitoring:** This approach constantly monitors fluctuations in the power consumption of the PCB. It allows for in-field measurements and online monitoring.
3. **High accuracy:** False positive rate (FPR) can be reduced to virtually zero, depending on desired detection threshold.
4. **No interruption or performance overheads to the original performance of the circuit under monitoring:** The approach continuously monitors the PCB without interrupting the performance or affecting its throughput.

A similar approach for IC HT detection has been applied by Zhang et al. [144] with the aim of HT detection prior to in-field use of the device. A current sensing resistor and an oscilloscope have been used to manually compute power consumption and detect the HT inside an FPGA. In our research, a PCB integrated run-time HT detection framework has been implemented, where every legitimate IC has a dedicated built-in or externally added power sensor computing the IC's power consumption.

Current integration method for detecting HTs inside ICs has also been suggested by Wang et al. [145]. Here the current consumption has been computed with a local sensor inside an IC. Using the current integration method, the data has subsequently been processed to detect HT induced anomalies. In our method, the difference between the readings from a global power consumption sensor and the sum of local power consumption sensors has been computed. This allows detection of hardware Trojans implanted on the PDN, located between the power source of the PCB and the legitimate ICs.

The approach proposed in this chapter has been validated with experimental measurements. The results show that HT detection can be achieved with a low FPR while keeping the performance degradation at a negligible level and the area overhead minimal. Note that the end-users are required to carry out similar analysis based on their golden

Figure 3.1: (a) Silicon die inside the package, (b) Sub Power Monitor and the die integrated into one package.

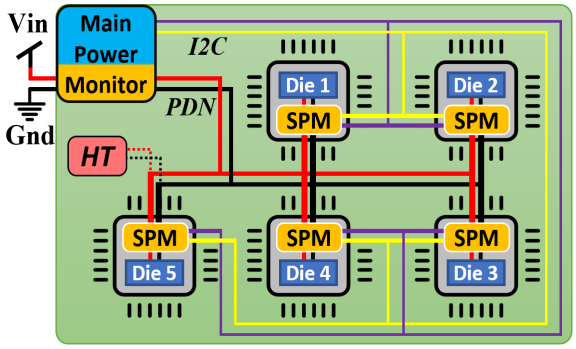


Figure 3.2: Block structure of a PCB with the proposed Differential Power Monitoring system and a hardware Trojan.

model and specific software workload scenarios. Similar to the case studies discussed in section 3.6, detection thresholds can be decided using the guidelines provided later in this chapter.

3.4 Proposed Methodology

Regardless of the payload, trigger and the amplitude of the damage caused, one common feature of additional components introduced to a circuit is an increase in power consumption. The proposed DPM method is designed to detect extra power usage on the board. For a DPM to be feasible, the individual chips on the PCB should be equipped with power consumption sensors, henceforth referred to as Sub Power Monitors (SPMs) (Figure 3.1). The monitoring circuit consists of a main power monitor (MPM) device on the main power rail, and SPMs integrated with individual chips on the PCB (Figure 3.2). The MPM includes a power sensor and a microcontroller for the noise dampening and data

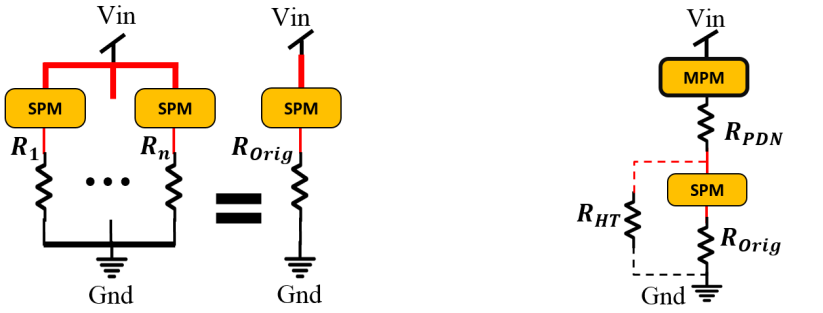


Figure 3.3: Abstraction of the (a) effective resistance of the original circuit, and (b) PCB with an added hardware Trojan.

communication logic. The dampening logic takes the moving average of the readings from power sensors, while the communication logic acts as the device user interface.

As illustrated in Figure 3.3a, the resistance of the original circuit on the PCB is formed by a parallel connection of multiple legitimate ICs. It can be summed up under one effective resistance (R_{Orig}). An HT device can be modelled as a resistive load (R_{HT}) added in parallel to R_{Orig} (Figure 3.3b). When the HT is non-operational, it consumes little power and the value of its effective resistance is high. However, when the HT is triggered, its power consumption increases and this can be modelled by decreasing the value of R_{HT}.

In the case of an ideal power distribution network (PDN), it can be assumed that its power dissipation (P_{PDN}) on parasitic resistance is zero. If every component on the PCB has an integrated SPM, the mismatch (ΔP) between the reading from the MPM (P_{MPM}) and the sum of the readings from SPMs ($\sum_{i=1}^{n}(P_{SPM})_i$) should be zero:

$$\begin{cases} P_{MPM} = \sum_{i=1}^{n}(P_{SPM})_i + P_{PDN} \\ P_{PDN} = 0, \end{cases}, \tag{3.1}$$

hence

$$\Delta P = P_{MPM} - \sum_{i=1}^{n}(P_{SPM})_i = 0. \tag{3.2}$$

However, real PDNs will naturally consume some power due to their non-zero parasitic resistance ($P_{PDN} \neq 0$). In addition, to account for the background noise, an extra term δ

should be introduced into Eq. (3.2) and ΔP will be given as follows

$$\Delta P = P_{PDN} + \delta. \tag{3.3}$$

Furthermore, since every device has its dynamic and static power consumption, adding an HT component to the circuit will increase ΔP. The final sought after variable is

$$\Delta P_{HTinf} = P_{MPM} - \sum_{i=1}^{n}(P_{SPM})_i = \Delta P + P_{HT}^{Dyn} + P_{HT}^{Stat}, \tag{3.4}$$

where ΔP_{HTinf} is the mismatch between the total power consumption (P_{MPM}) and the sum of power consumptions by individual chips ($\sum_{i=1}^{n}(P_{SPM})_i$) on an HT infected PCB. The HT's dynamic and static power consumptions are represented by P_{HT}^{Dyn} and P_{HT}^{Stat}. It is possible to detect a triggered HT through continuous monitoring, if, for a given PCB layout, the maximum value of the power distribution network's parasitic power consumption $max(P_{PDN})$ (e.g. due to resistive, capacitive and inductive components) is empirically known. Once the HT triggering mechanism is activated, the internal states of the HT chip will switch to deliver the malicious payload. This inevitably introduces a sharp increase in the power consumption of the extra component represented as dynamic power consumption P_{HT}^{Dyn} in Eq. (3.4). In turn, the spike in P_{HT}^{Dyn} translates into a spike in ΔP_{HTinf} values (Figure 3.4).

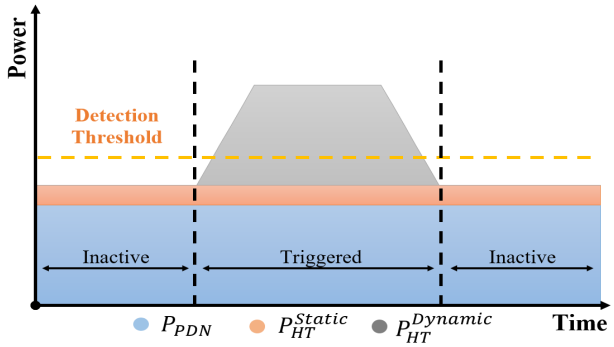


Figure 3.4: Change in ΔP_{HTinf} when the HT is triggered.

In order to detect the HT based on increased ΔP_{HTinf} values, a new parameter is introduced - detection threshold (Figure 3.4, Figure 3.5). The choice of the detection threshold for a particular PCB is based on a number of parameters including maximum

acceptable False Positive Rate and minimum detectable power consumption of the hardware Trojan. Note that the detection threshold defines the highest value of ΔP_{HTinf}, above which anything is flagged as an active HT on the board. Its value is pre-programmed in the MPM block, taking into account the factors mentioned above.

In addition, along with the rise in the value of P_{MPM}, another characteristic feature of a triggered HT is the drop in $\sum_{i=1}^{n}(P_{SPM})_i$ which can be described by the following expression:

$$\Delta\sum_{i=1}^{n}(P_{SPM})_i = \sum_{i=1}^{n}(P_{SPM})_i - \left[\sum_{i=1}^{n}(P_{SPM})_i\right]_{Triggered}$$
$$= \frac{{V_{in}}^2}{R_{Orig}}\left[\left(\frac{R_{Orig}}{R_{PDN}+R_{Orig}}\right)^2 - \left(\frac{R_{new}}{R_{PDN}+R_{new}}\right)^2\right], \quad (3.5)$$

where $R_{new} = R_{Orig}R_{HT}/(R_{Orig}+R_{HT})$ and V_{in} is the input voltage. The power drop in Eq. (3.5), whose behaviour is schematically shown in Figure 3.5, is due to the increased voltage drop on the PDN, after the hardware Trojan has been triggered.

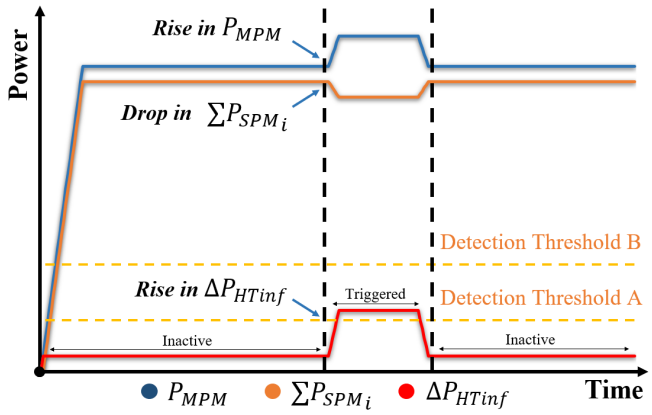


Figure 3.5: Change in power consumption pattern when a hardware Trojan is triggered.

The dependence of the drop in the combined SPM power consumption on R_{HT} is shown in Figure 3.6 for three values of R_{PDN}. In this work, we have primarily focused on the spikes in ΔP_{HTinf} values, which take into account both the effect of the drop in $\sum_{i=1}^{n}(P_{SPM})_i$ and the rise in P_{MPM}, since ΔP_{HTinf} is the difference between them.

In order to reduce the background noise δ in Eq. (3.3), filtering by the moving average of every recorded ΔP_{HTinf} with its previous $N-1$ values is applied, where N (averaging level) is the number of averaged points. The resulting values are stored in a complementary

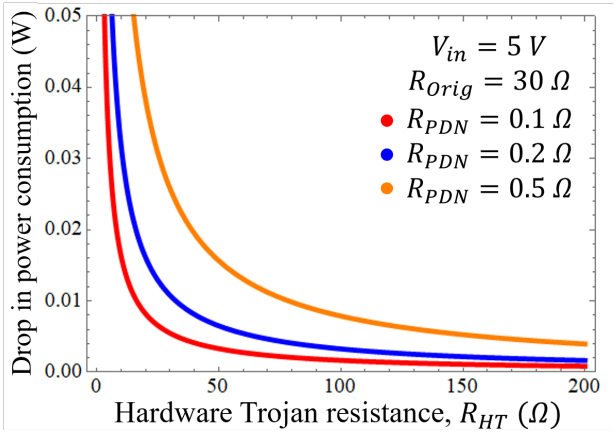


Figure 3.6: The effect of R_{HT} on the drop in the registered combined Sub Power Monitor power consumption $\sum_{i=1}^{n}(P_{SPM})_i$.

$\Delta P_{HTinf}^{(N)}$ variable defined as

$$\Delta P_{HTinf}^{(N)} = \frac{\sum_{i=1}^{N}(P_{HTinf})_i}{N}.$$

Detectability trade-off between the HT activation time and power consumption should be considered since a higher averaging level N will result in a loss of accuracy in detecting short duration HT payloads, but will allow for detection of low-power HTs. The application specific optimal value for the averaging level N can be estimated from the following inequality

$$t_{HT} \geq N\frac{1}{\nu}, \tag{3.6}$$

where t_{HT} is minimum detectable HT activation time given as a technical requirement, and ν is the recording frequency of the monitoring system.

This technique successfully dampens the distortions introduced by the noise δ, as well as anomalies introduced through sensor error, which would otherwise be interpreted as peaks induced by an active HT (section 3.6).

Detection Range Analysis

The detection threshold P_{thr} can be calculated using the following formula

$$\begin{cases} P_{thr} = \alpha \dfrac{|\delta_{\max}|}{N} + P_{PDN} \\ P_{SensRes} \leq P_{thr} \leq P_{HT}, \end{cases} \tag{3.7}$$

where N is the averaging level, P_{PDN} is the parasitic power consumption of the power distribution network, $P_{SensRes}$ is the power sensor resolution, and P_{HT} is the minimum value of the HT power consumption that we aim to detect. The coefficient α ($\alpha \in (0, 1]$) can be determined experimentally on the given setup, once the value of the desired FPR has been chosen. For example, if it is required to have $FPR = 0\%$, then $\alpha = 1$. As for δ_{max}, it represents the maximum value of the background noise δ, which can be approximated by a normal distribution with zero mean. It should be noted that the threshold is determined as a function of the amplitude of noise, the averaging level, and the power sensor resolution.

If n is the number of independent DPM outputs f_i ($i = 1, ..., n$), each of which has HT detection probability p, then f_i can be described as a Bernoulli distribution [146] with

$$f_i = \begin{cases} 0 & \text{HT not detected} \quad \Delta P_{HTinf} < P_{thr} \\ 1 & \text{HT detected} \qquad \Delta P_{HTinf} \geq P_{thr}, \end{cases} \tag{3.8}$$

where ΔP_{HTinf} is defined in Eq. (3.4). It follows from (3.8) that the HT is detected when ΔP_{HTinf} reaches the HT detection threshold Eq. (3.7). If there is an active HT on a PCB, the maximum likelihood estimator for detection probability p for Bernoulli distribution Eq. (3.8) can be calculated as follows [146]

$$p = \frac{\sum_{i=1}^{n} f_i}{n}. \tag{3.9}$$

The detection probability Eq. (3.9) is proportional to the observed DPM outputs resulted in detection of the HT. It can be seen from (3.8) that this probability can be increased by decreasing P_{thr}, i.e. the detection probability is a function of the same parameters as P_{thr}.

3.5 Experimental Setup - Prototype PCB

The original, hardware Trojan free, circuit consists of four 16 MHz ATmega328P microcontrollers which have been programmed and wired up into four blocks: authentication (*Auth. 1, Auth. 2*), processing, and memory (Figure 3.7, Figure 3.8). Additionally, an Inter-Integrated Circuit (I^2C) communication line multiplexer, a keyboard and two displays have been integrated into the setup.

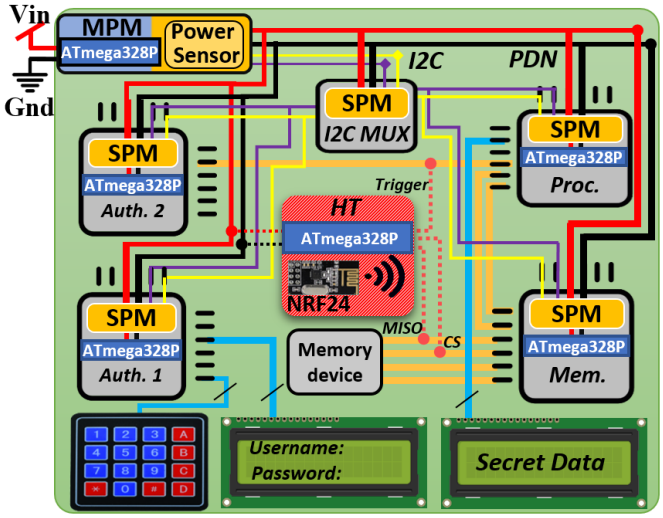


Figure 3.7: Diagram view of the PCB prototype.

The general function of the system is secret data transmission from the memory block to one of the two integrated displays. Next, three different HT devices (cases 1-3) have been introduced into the original circuit with the malicious purpose of data leakage. Finally, the proposed run-time Differential Power Monitoring circuit has been implemented on the setup.

3.5.1 The Original Circuit

The function of the original circuit (Figure 3.8, shown by an orange border) is to store and display data from the built-in memory block, after a log-in procedure. A keyboard and a display, linked to the authentication block, are leveraged to facilitate the log-in process. Upon a successful log-in event, an enable signal is generated by the authentication block. This signal is fed into the processing block. The enable signal triggers the processing

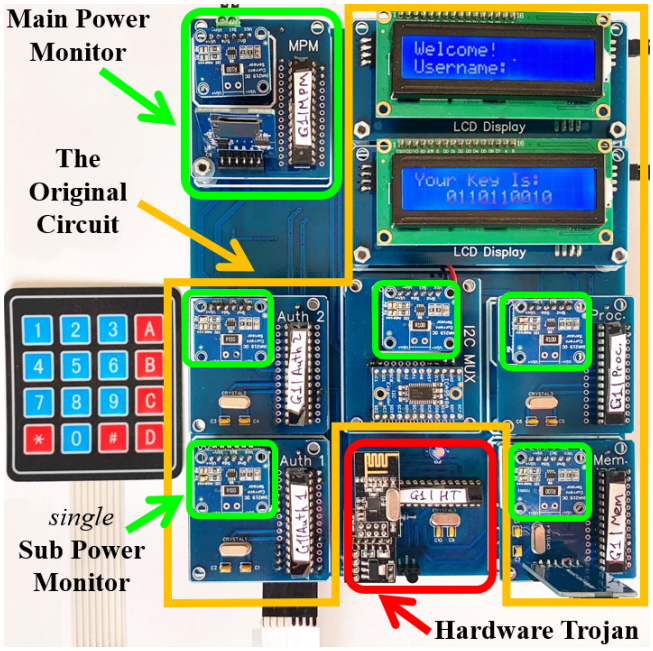


Figure 3.8: Real life photograph of the PCB prototype.

block to fetch the secret data from the memory block. The processing block uses Inter-Integrated Circuit (I^2C) communication protocol to request from the memory block. In turn, the memory block requests the data from a built in memory device through Serial Peripheral Interface (SPI). Once the secret data is passed to the processing block, it is presented on a second display. The communication with both displays is executed through I^2C protocol. After a pre-set time, the system automatically clears the displays and logs out, getting ready for a new log-in cycle.

3.5.2 The Hardware Trojan Device

3.5.2.1 Hardware Trojan Case 1

In this attack scenario the HT has three components: a 16 MHz ATmega328P microcontroller, an NRF24L01 System-on-Chip 2.4 GHz radio frequency transceiver and a 5 V to 3.3 V logic level shifter (shown in Figure 3.7 and Figure 3.8 with a red border). The device has five I/O pins: two power input and three data wires for timing the attack and eavesdropping on the interconnections on the original circuit. Two of the data wires are tapped to the SPI interconnections in the memory block. One of these wires carries

a trigger signal, which turns the HT into an active transmitter, while the second wire is used to read the secret data. Additionally, a third wire is linked to the enable signal from the authentication block. By default, the HT is set to sleeping mode to minimise the power consumption to stay undetected, and it only wakes up on a log-in event. To further increase its stealthiness, it is programmed to leak data after a certain number of log-in events. When the HT has reached the data leakage iteration, it powers on the radio-transmitter, which is otherwise turned off in order to reduce power consumption and avoid detection. Finally, the data is leaked out, and the HT device is sent back to sleep.

3.5.2.2 Hardware Trojan Case 2

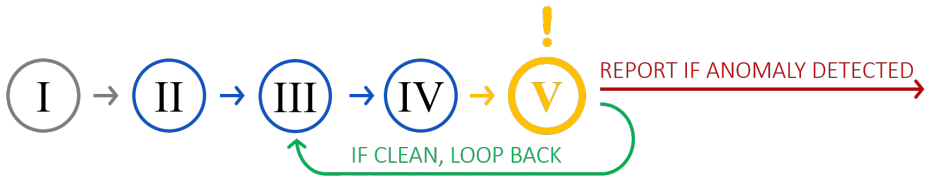
To reduce the power consumption, we now consider an HT device with a 16 MHz ATmega328P microcontroller unit. The HT has two states: 1) a state of active payload delivery after triggering, and 2) an idle state. The power consumption of the device in state 1 is around 5 $mW - 10\ mW$, whereas in state 2 it is under 300 μW (Power-Down mode). In case 2, the payload of the HT is assumed to be arbitrary. The condition-based activation trigger for the HT can be either the enabling signal from the PCB circuit similar to case 1, or an internal watchdog timer. The HT's pin connections are also executed in a similar way to case 1.

3.5.2.3 Hardware Trojan Case 3

The always-on HT discussed in case 3 is similar to the one in case 2. A major difference, however, is in the operating pattern. As opposed to case 2, here the HT does not have a triggering condition and is directly set to the only available active state after power up of the PCB. The power consumption of the device is in the range of 5 $mW - 10\ mW$. In case 3 the payload of the HT is assumed to be arbitrary. The pin connections of the HT, except for the trigger signal pin, are executed in a similar way to that in case 1.

3.5.3 The Power Monitoring Circuit

There are two blocks to the power monitoring circuit: (a) a Main Power Monitor (MPM), and (b) five Sub Power Monitors (SPM) (Figure 3.2, Figure 3.7, Figure 3.8 marked in green borders). Inside the MPM, along with the power sensor, an ATmega328P



I. Start: Power up the PCB.
II. MPM sets up all power sensors on the PCB.
III. MPM receives power readings from all SPMs.
IV. MPM processes the gained information.
V. Perform the prediction and loop to step three.

Figure 3.9: Flow chart of Differential Power Monitoring logic steps.

microcontroller has been used to conduct the on-board data processing and to deliver the dampening and communication functions. The microcontroller is linked to all power sensors via a multiplexer for communication using the I^2C protocol. As illustrated in Figure 3.9, after the initial power up (step I), the microcontroller in MPM sets up all the power sensors on the PCB (step II). Next, in step III the MPM requests and receives power readings from all the sensors at a given time. Note that within one iteration the time-span between any two SPM readings is negligible. Then, in step IV it processes the obtained power readings. Finally, it follows from Eq. (3.8) that if the difference is larger than the pre-set detection threshold, an HT detection alarm is raised and the microcontroller begins the next iteration (step V). These steps are described in more detail in the form of the pseudo-code Algorithm 1. Power consumption measurements are taken with INA219 high-side current and power monitor chips. These chips have a built in 12 bit ADC and provide readings with a resolution of 1 mW.

3.6 Experimental Results

The performed experiments support the theory described in section 3.4. As a proof of concept, the hardware prototype has been tested against three different HT attacks for detection capability assessment. Additionally, an HT detectability analysis has been carried out by dampening the noise level and finding the corresponding lowest threshold with 0% False Positive Rate (FPR). An experiment has also been set up to address

Algorithm 1 Main Power Monitor data flow.

▷ **Step I**
1: $P_{thr} := DetectionThreshold$ ▷ *Set Detection Threshold*
2: $N := AveragingLevel$ ▷ *Set Averaging level*
3: $\Delta P_{MovAvg}[\Delta P, N] := 0$ ▷ *Initialise moving average*
4: **Initiate power sensors**. ▷ **Step II**
5: **while** 1 **do**
▷ *Storing data from sensors* | **Step III**
6: $\mathbf{P_{MPM}} := sensor[0].GetPower$
7: **for** $i = 1, 5$ **do**
8: $\mathbf{P_{sumSPM}} += sensor[i].GetPower$
9: **end for**
▷ *Find new ΔP* | **Step IV**
10: $\mathbf{\Delta P_{new}} := P_{MPM} - P_{sumSPM}$
▷ *Update the moving average with new ΔP*
11: $\Delta P_{MovAvg}[\Delta P, N] := \Delta P_{MovAvg}[\Delta P_{new}, N]$
12: **if** $\Delta P_{MovAvg}[\Delta P, N] > P_{thr}$ **then** ▷ **Step V**
13: *Hardware Trojan detected.*
14: **end if**
15: **end while**

varying and complex workload situations, which can induce an abrupt change in the power consumption of the legitimate board-level components. The purpose of this experiment is to validate the detectability of an HT on the PCB in a scenario, where the legitimate ICs are running with varying workloads.

3.6.1 Proof of Concept and Averaging-Threshold Tradeoff

Data obtained from the experiments confirm the capability of the proposed DPM method to detect the presence of an HT on the PCB. The power values in Figures 3.10-3.12 and Figures 3.14-3.16, all functions of time, have been recorded with frequency ν of the monitoring setup and presented as time series, e.g.

$$\{\Delta P(t),\ t = \frac{n}{\nu},\ \nu = 100\ Hz,\ n = 1, 2, 3, ...\}. \tag{3.10}$$

All experiments in this subsection have been carried out at room temperature (20°C).

HT Case 1: As illustrated in Figure 3.10, there is a clear spike in ΔP_{HTinf} value, when an HT is triggered. If the value of this spike is larger than the pre-defined detection

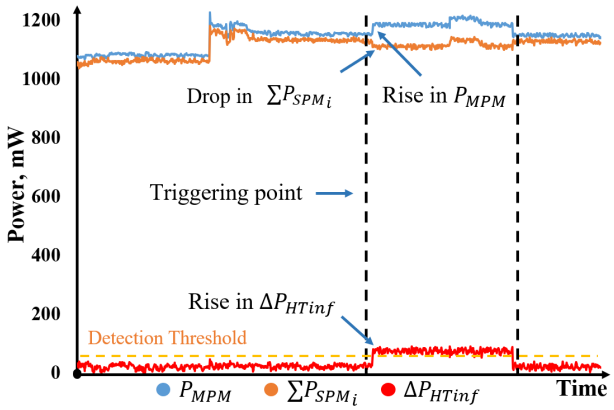


Figure 3.10: Spike in ΔP_{HTinf} when HT is triggered, and drop in $\sum_{i=1}^{n}(P_{SPM})_i$ alongside the rise of P_{MPM} (HT case 1).

threshold, the HT will be detected. Note that as the detection threshold is lowered, lower power HTs become detectable. Two different detection thresholds are illustrated in Figure 3.11. Here, level A provides a better HT detection resolution than level B, since it is capable of detecting lower ΔP_{HTinf} spikes. However, by lowering the detection threshold from B to A, the False Positive Rate increases due to the noise on ΔP_{HTinf} values.

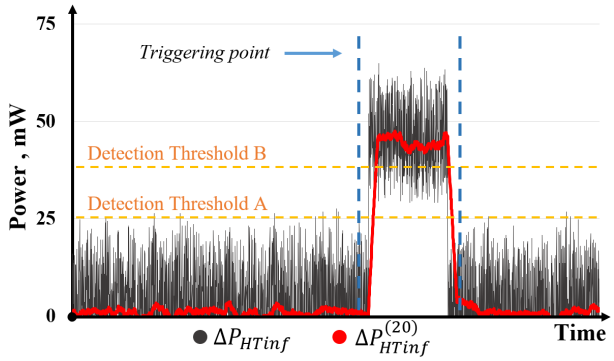


Figure 3.11: Change in ΔP_{HTinf} and $\Delta P_{HTinf}^{(20)}$ during HT triggering (HT case 1).

Post-processing of the results of the measurements shows that applying a moving average filter greatly reduces the impact of noise on the raw ΔP_{HTinf} values (Eq. (3.4)) as illustrated in Figure 3.11 and Figure 3.12. As can be seen in both figures, the $\Delta P_{HTinf}^{(20)}$ values produce a smoother signal with the averaging parameter set at $N = 20$.

One downside to this averaging technique is the risk of losing data on HT short-time scale activation, since the data may be smoothed in the same way as an anomalous peak. This issue can be addressed by introducing a higher frequency monitoring system,

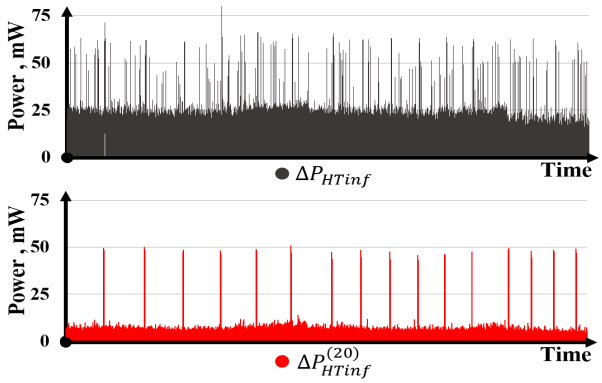


Figure 3.12: Raw ΔP_{HTinf} (top graph), and $\Delta P^{(20)}_{HTinf}$ (bottom graph) with an averaging of 20 (HT case 1).

which is fast enough to record data several times while the HT is active. Ideally the monitoring system will sample the power readings of the PCB at least N times while the Trojan is active, where N is the number of data points used to calculate the moving average. The optimal value of N can be achieved through experimental means. On the other hand, the use of an averaging function creates a less noisy signal. As it can be seen in Figure 3.11 the individual values of a noisy signal (black line) can be notably deviated from their average shown by the red line. By taking the moving average, these deviations are mostly alleviated. This allows us to detect an HT device drawing a lower amount of power. In addition, the moving average filtering provides significant improvements to the False Positive Rate (FPR) and the same values of FPR are reached at notably lower detection thresholds (Figure 3.13). For example, with the experimental setup under test, over thirty-fold reduction of detection threshold has been recorded for an FPR of zero percent, when the raw ΔP_{HTinf} (63 mW) values are compared to their moving averages $\Delta P^{(50)}_{HTinf}$ (1.9 mW) (Table 3.1). As shown in Table 3.1, the minimum detection threshold for 0% FPR consistently drops along with the increase of the averaging level.

Table 3.1: Detection thresholds for $FPR \approx 0\%$

Time series	ΔP_{HTinf}	$\Delta P^{(5)}_{HTinf}$	$\Delta P^{(20)}_{HTinf}$	$\Delta P^{(50)}_{HTinf}$
Threshold	63 mW	20 mW	3.5 mW	1.9 mW

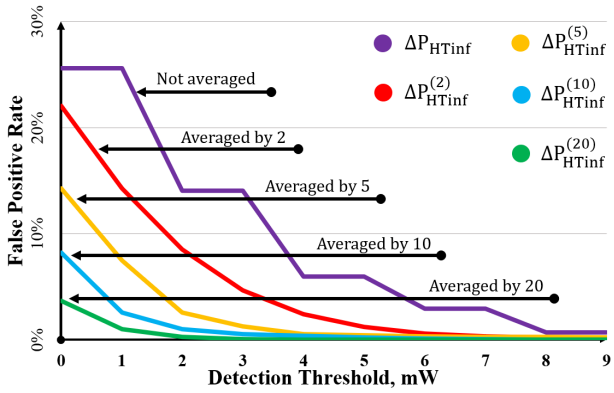


Figure 3.13: False Positive Rate for 5 levels of averaging of ΔP_{HTinf} (HT case 1).

HT Case 2: In this case, described in section 3.5, a conditionally activated HT has been used for the attack. The results show that with a prior knowledge of the appropriate detection threshold (4 mW) the hardware Trojan is easily detectable. As illustrated in Figure 3.14, the power consumption of the activated HT (red line) fluctuates between 6 mW and 9 mW. It can also be seen by the green line that the $\Delta P^{(20)}$ values from Eq. (3.10) fluctuate around 2 mW. The experiment has been repeated on a batch of 5 PCBs and the worst-case scenario PCB with the highest value of detection threshold for $FPR \approx 0\%$ has been chosen for illustration.

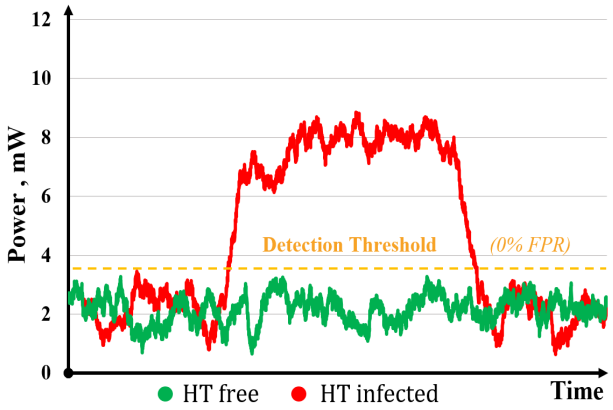


Figure 3.14: Detection of a triggered hardware Trojan with $\Delta P^{(20)}_{HTinf}$ (HT case 2).

HT Case 3: In this case, an always-on HT has been used for the attack. The HT turns on when the PCB is powered on. As can be seen in Figure 3.15, the moving average of the HT's power consumption (red line) was around 7.5 mW. Shown in green is the baseline

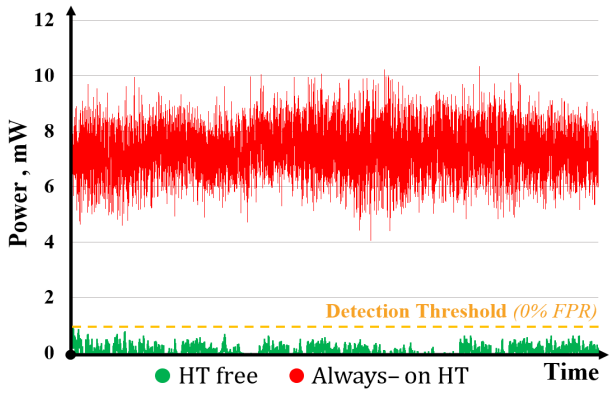


Figure 3.15: Detection of an always-on hardware Trojan with $\Delta P^{(75)}_{HTinf}$ (HT case 3).

PDN power consumption based on characterisation of a batch of HT free PCBs. It can be seen that by using a predefined detection threshold of 1 mW it is possible to detect the HT with near 0% FPR. It should be noted that if the HT activation time is very long, as in the always-on case, the averaging level can be set to a high number (e.g. $N = 1000$) which will filter the noise down to the order of tens of μ-Watts. Hence, the HT power detection threshold Eq. (3.7) can be lowered to the same order.

Note that the proposed method uses different moving average filters to detect HTs with different characteristics (e.g. power consumption, active time). Depending on the targeted HT characteristics, one or more moving average filters may be deployed for optimal HT detection.

Variable Workload of Legitimate ICs: More complex situations with variable workloads of legitimate on-board components have been considered in this experiment. The Differential Power Monitoring method is used to detect the HT which has been programmed to switch on and off with a given time period and a consequent switch in power consumption from 7 mW to less than 300 μW. As can be seen from Fig. 3.16, four legitimate ICs (P_{SPM1}-P_{SPM4}) change their power consumption at different times. This change is also clearly visible in the overall PCB power consumption (P_{MPM}). The results show that the power consumption of the HT (ΔP_{HTinf}) has been detected. As expected, it has repeatedly exceeded the predefined detection threshold despite the changes in the power consumption of the legitimate ICs.

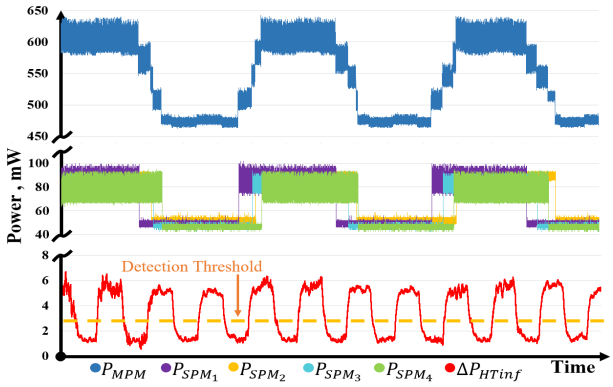


Figure 3.16: Detection of a hardware Trojan under variable workload of legitimate components (HT case 2).

3.6.2 Possible Attacks and Countermeasures

To further improve the proposed methodology it is crucial to understand how it can be circumvented. Two possible scenarios of an HT attack can be considered.

In the first scenario the adversary targets one of the legitimate IC and SPM module pairs. The communication wires between this victim SPM and the MPM module, as well as the power source wires of the legitimate IC are cut. Next, an HT with its dedicated fake SPM module is added under the same address name. Finally, both the HT and legitimate IC power are sourced through the fake SPM and it is connected to the MPM module. Now, whenever the MPM tries to read the victim SPM, it will actually address the fake SPM. This way the HT power consumption is added to the overall SPM measurements and is thus not detected. This attack can be counteracted by introducing existing HT prevention methods. For example, using cryptography [142] for the communication between the MPM and SPM modules can prevent the attack described above.

In the second scenario the HT is not powered from the power distribution network. An example of such an attack, where the HT is powered from an I/O pin of a legitimate IC, is mentioned in section 3.2. In this case the extra power consumption of the HT will be attributed to the legitimate IC and not contribute to ΔP_{HTinf}. Thus the HT will be invisible for the proposed monitoring system. Developing an algorithm based on characterisation of a batch of a given PCB design will help detect deviations from the expected power consumption pattern for each legitimate IC. Such deviations will indicate

the presence of an HT device on the PCB. This approach will also address the first attack scenario discussed above.

3.7 Concluding Remarks

This work is the first to develop a methodology on detecting hardware Trojan (HT) components on a Printed Circuit Board (PCB) using power monitoring. The methodology proposed here is independent of the HT's trigger and payload functions. The presented results show that the proposed Differential Power Monitoring (DPM) method, based on power consumption, can detect HT devices implanted on the PCB, with a false positive rate (FPR) that can become zero by selecting appropriate detection threshold (Table 3.1).

The Differential Power Monitoring technique provides additional protection for end users without affecting the throughput of the PCB. It can be employed in conjunction with other PCB HT countermeasures without cross-disruption. Key variables in the DPM technique that can be improved upon are sensor frequency, resolution and accuracy, and the averaging level for ΔP_{HTinf}. The proposed methodology can be further improved by using more sophisticated sensors, communication protocols and carrying out theoretical underpinning.

Thus, the objectives set out in section 2.7 concerning the first contribution have been met in this chapter. In the following chapter, the proposed monitoring method is further developed, using machine learning algorithms to detect HTs powered from I/O ports of legitimate chips on the PCB. Two ML algorithms, One-Class Support Vector Machine (SVM) and Local Outlier Factor (LOF), have been applied on the power consumption data harvested experimentally from a purpose built PCB prototype.

Chapter 4

PCB Hardware Trojan Run-time Detection Through Machine Learning

In this chapter two machine learning (ML) methods have been applied to detect HTs running on power from I/Os of legitimate chips on a PCB. A PCB prototype has been fabricated to obtain real-life data, which was used to train two ML algorithms: One-Class Support Vector Machine and Local Outlier Factor. For validation of the ML classifiers, one hundred categories of HT devices have been modelled and inserted into the validation and testing datasets. Simulation results show that using the proposed methodology an HT device can be detected with high prediction accuracy (F1-score above 99.7% for a 50 mW HT). Further, the ML model has been uploaded to the prototype PCB for hard-silicon validation of the methodology. To the best of our knowledge, this is the first work on real-time detection of PCB HTs, which are powered from the I/O pins of legitimate ICs. Experimental results show that the performance of the ML model on a real-life prototype is consistent with that of the simulations.

4.1 Introduction

In the last few years several PCB assurance approaches have been developed which can be categorised as in-circuit testing, functional testing, JTAG boundary scan, bare-board testing and visual inspection [147]. With hardware attacks becoming stealthier, finding more advanced detection methods has now become crucial. Particularly important are automated quality assurance (AQA) methods, which being non-destructive can detect a wide range of HTs on a PCB with minimal human involvement [147]. One of these methods is automated visual inspection (AVI) for PCB verification and authentication. This method

is about testing on assembly lines during manufacturing, replacing human oversight, thus preventing unauthorised modifications during assembly. This PCB AQA approach can be used in different stages of the manufacturing cycle and can be categorised by the imaging modalities of choice and image analysis techniques [122]. Several AVI have been suggested including canonical image processing methods for detecting trace and via level defects [123], convolutional neural network for detecting known defects [124], automated detection for component misplacement by directly comparing golden and test PCBs [125], [126] and text detection on the PCB for verification purposes [127], [128]. Although AVI has so far been the most commonly used method for PCB assurance it has limitations such as absence of run-time monitoring option and demand for significant subject-matter expert involvement [148], [149].

Conventional PCB quality assurance techniques have not been inherently designed for detecting deliberate malicious alterations. Advancements in state-of-the-art technology, possibly used by adversaries creating stealthier HTs, introduces further strain on legacy AQA techniques. Novel machine learning (ML) based techniques, on the other hand, provide a fresh perspective on effective countermeasures for detecting HTs on PCBs [147], [150]. For instance, the methodology proposed in [141] detects both, always active and triggerable HTs on the PCB. Using a distributed power sensing architecture, it is experimentally demonstrated that the proposed method can detect HTs with power consumption as low as 7.5 mW, powered from the power distribution network, similar to Trojan A in Figure 4.1. However, this monitoring system would be oblivious to any

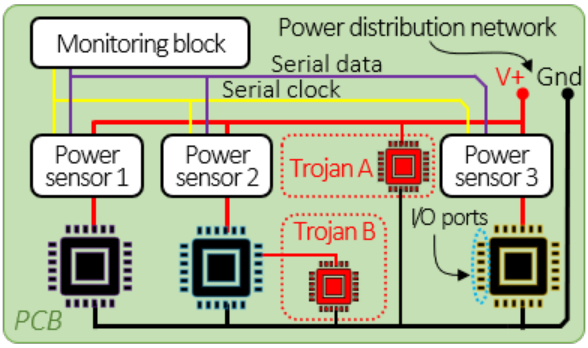


Figure 4.1: PCB diagram with proposed power monitoring circuit, Trojan A powered from the power distribution network and Trojan B powered from I/O port of a legitimate chip.

HT that is powered from the input/output (I/O) pins of the legitimate integrated circuits (ICs), similar to Trojan B in Figure 4.1. The main novelty in this chapter is using ML algorithms implemented on the monitoring block, combined with a similar monitoring circuit architecture (Figure 4.1) as proposed in [141], to detect HTs powered from the I/Os of legitimate ICs on the PCB. Using ML to inspect power consumption for HT detection, as proposed in current work, is readily scalable, easy to deploy on a given PCB design and much less intrusive than other approaches such as impedance anomaly detection, requiring no extra actions taken by the ICs on the existing circuitry.

The proposed monitoring setup should be designed and manufactured alongside the original PCB circuitry. The ML model training should be done on data from a complete simulation model or, preferably, group of prototype PCB devices, prior to large scale manufacturing. To ensure the integrity of the HT-clean power consumption training data, the batch of prototype PCBs can be subjected to reverse-engineering testing or expert visual inspection. Both methods are used to test the golden PCBs for HT presence, but have their own shortcomings in volume manufacturing.

Prior Work

Machine learning techniques have a huge potential for addressing countermeasures for various HT attacks [89], [90]. Such methods have already been widely applied for IC HT detection (section 2.5.2), improving the detection capabilities in a number of aspects including reverse engineering [92], side-channel analysis [93], [94], real-time detection and for HT detection on gate-level netlists [95]. A comprehensive survey of the latest research and developments of ML-based approaches for HT detection and prevention on an IC level is carried out in [96], where it has been demonstrated that the number of research publications and achievements in IC HT detection using ML techniques have been growing rapidly since 2014. This shows that ML is a promising tool for guaranteeing security of hardware.

HT vulnerabilities for PCBs are not too different to those at the IC level. Hence the ML methods applied for countermeasures against HTs at the IC level can be useful in HT detection and prevention at PCB level. While ML algorithms have proved to be highly promising in overcoming IC HT threats, ML based techniques for higher levels of system

abstraction such as PCBs have not been well studied. That said, some research has very recently become available. Pearce et al. [151] have proposed an HT detection methodology utilising multiple side-channel performances, including power consumption, temperature and electromagnetic field as well as communication and CPU activity patterns. In their approach they use defender controlled operating environments to have multiple views on the performance of the PCB, hence a higher chance to detect the presence of an HT. Another promising approach is taken in [152], [153], [154] and [155], where the key HT detecting factor is the alteration to the expected impedance value of either the power distribution network (PDN) or the metal connections between the legitimate ICs. In case of a sufficiently high resolution this approach is capable of detecting minor alterations to the circuit.

The rest of the chapter is organised as follows: The attacker model and the proposed approach are described in section 4.2. In section 4.3 theoretical background is provided on two ML algorithms used in this work. Section 4.4 explains the proposed methodology, while section 4.5 describes the experimental setup, including the prototype PCB. Section 4.6 discusses simulation results. Further, real-life experimental results are given in section 4.7. Discussion around the proposed methodology is provided in section 4.8. Finally, the chapter is concluded in section 4.9.

4.2 Attacker Model and Our Approach

When added to the original PCB circuitry, an HT device can draw power from sources such as the mains supply, a built in battery, capacitor or energy harvester, the power distribution network or an I/O pin (or trace) of a legitimate IC. A thoughtful adversary would try to conceal the modifications in the internal layers of the PCB to avoid detection. If the HT is powered from the mains supply, it would need external vias and traces for linkage. Similarly, a battery, a capacitor or energy harvester would be too large to be concealed in the internal layers. In both of these cases an automated visual inspection can be sufficient to detect any external modifications on the device, thus ultimately leading to the detection of the HT. On the other hand, a carefully designed HT

can completely escape detection from visual means of inspection, if it is powered from the power distribution network or an I/O trace of a legitimate IC. The first of these cases has already been addressed in [141], where the suggested differential power monitoring method demonstrated sufficiently high accuracy in detecting an HT which consumes power from the power distribution network. This work addresses this problem by proposing a methodology for detecting such HTs using machine learning.

The main characteristics of the proposed approach are:

1) **Trojan payload invariant:** The proposed methodology is independent from the types of payload and trigger mechanisms of the HT device, assuming it consumes power.

2) **Real-time monitoring:** Continuous monitoring of the power consumption patterns on the PCB in real-time, searching for anomalies.

3) **High prediction quality:** Classification F1-score can reach over 99.5%, depending on the HT parameters (e.g. power consumption, active time).

4) **No interference with the performance of the original circuit:** The approach is completely detached from the computational provisions of the circuit under inspection, continuously running its algorithms without degrading the performance or affecting the throughput of the PCB.

4.2.1 Assumptions

It is assumed that the adversary can use any HT regardless of its payload and trigger, as long as the activation of the HT alters the power consumption of the original circuitry. It is also assumed that the IC and PCB design houses, as well as the firmware and intellectual properties for designing the ICs are trusted. The only possible threat comes from outsourcing PCB production to untrusted facilities, or during the transfer of the product from the manufacturer to the consumer.

There is also a possibility of adversaries swapping a trusted IC on the PCB with an HT infected IC. This attack scenario is out of the scope of this work since it is concerned with the research of IC level HTs, which discusses modifications of ICs during their design and production phases. The alternative case assuming the adversary has arranged their separate and unpoliced illegal IC production is unlikely given the prohibitive financial constraints on organising silicon chip production and limited access to original IC design

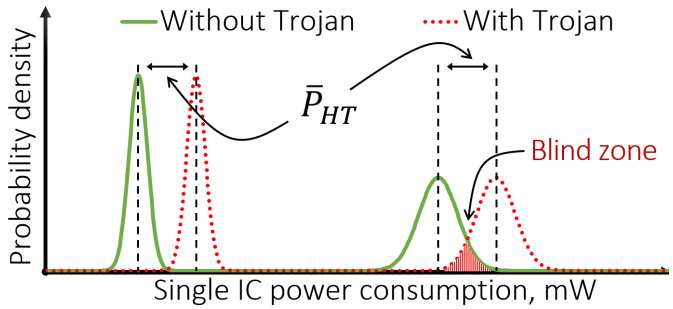
and layout information. Thus, it can be assumed that the ICs on the PCB are trusted and HT-clean.

4.2.2 Our Approach

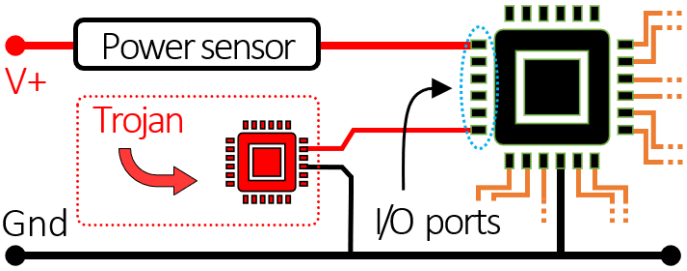
The main function of the proposed approach is to differentiate between two groups of power consumption data, HT-clean and HT-contaminated, incoming during run-time. The HT-clean data group belongs to the case, where no HT has been implanted on the PCB and the device is functioning as expected. This case includes all the tasks and modes that the device is expected to operate in with their respective power consumption patterns. Any deviation from the expected power performance is deemed to be a fault or an HT attack, and the corresponding data-point belongs to the HT-contaminated group. This second group, where all the HT-contaminated data-points reside, is harder to model since it must include all the possible HT attack scenarios. To tackle this problem, one class classification machine learning algorithms are used in this work for anomaly detection. The algorithms should be able to separate clean and contaminated data-points, having previously seen only the group with clean data-points. Such algorithms are commonly referred to as anomaly detection algorithms.

4.2.3 Blind Zone

Next, the definition of the term blind zone is given here. To simplify the problem at hand, let us consider a case with a single IC on the PCB. Let us further assume that this IC has two work modes: idle and operating. In that case it would be safe to assume that its power consumption probability density function would look like the green line in Figure 4.2a, with two distinct peaks, one for every work mode. When a Trojan device is added on the PCB, which consumes power from an I/O pin of the legitimate IC Figure 4.2b, the resulting new probability density function will be skewed to the right by the mean power consumption value of the HT. This new distribution is drawn in red in Figure 4.2a. Intersection of the regions under green and red functions are where it is not possible to confidently separate HT-clean and HT-contaminated cases. In these regions, referred to as blind zones, the HT-contaminated datapoints overlap with HT-clean datapoints. This effect primarily happens for two reasons. First, when the HT's power consumption



(a) Power consumption probability density functions



(b) Legitimate IC and an implanted Trojan schematic

Figure 4.2: Illustration of a single IC and single Trojan.

P_{HT} is so small that it almost does not affect the legitimate power consumption curve, making it very hard to detect. In Figure 4.2a it will mean that the two tall peaks and the two short peaks overlap. This is an inherent shortcoming of any side-channel analysis methodology, when the footprint on the side-channel is too small to detect. Second case is when the HT's power consumption is exactly large enough to skew the HT-contaminated (red) power consumption curve to the right, so that it overlaps with the second legitimate green peak. In Figure 4.2a it will mean that the taller red peak will overlap with the shorter green peak.

4.3 Anomaly Detection Classifiers

4.3.1 Anomaly Detection: Novelties and Outliers

Anomaly detection algorithms are divided into two machine learning method subgroups: novelty detection and outlier detection. While in novelty detection the training

data does not contain anomalies, which can only appear in new observations in the prediction stage, in outlier detection the training data does contain some anomalies. In the outlier detection method the algorithm finds the regions where the training data is mostly congregated, ignoring the observations which have deviated from the majority data. For the HT detection problem on PCB set out in this research novelty detection approach is most appropriate.

Two relevant ML algorithms for power consumption novelty detection are one-class classifiers based on One-Class Support Vector Machine (SVM) and Local Outlier Factor (LOF) implemented in Scikit-learn library [156]. One-Class SVM and LOF have different underlying mechanisms to produce the prediction, but both are suitable for supervised learning with only one of the classes initially present in the training dataset, i.e. HT clean PCB power consumption data. This is an important characteristic, since it would be unrealistic to assume the possibility of two class classification by modelling all possible HT attack scenarios the adversaries can come up with. Given the non-linear nature of the task assigned to the ML models in this project, i.e. one-class classification, radial basis function (RBF) kernel has been used with One-Class SVM algorithm, making its decision boundary very flexible to variations in the dataset. In other words, the radial basis function allows the ML model to closely replicate the correct power consumption pattern locations in a multi-dimensional space with high precision, where the number of dimensions are equal to the number of ICs being monitored.

4.3.2 One-Class Support Vector Machine

If the conventional SVM method separates different classes existing within a given dataset [157], One-Class SVM [158] minimises the radius of a hypersphere that encloses the training data consisting of only one class. This is achieved through a suitable kernel function, which maps feature vectors onto a higher dimensional feature space. The method defines an origin and finds a hyperplane, which separates the origin and the mapped feature space with a maximum margin. Thus, unlike multi-class SVM, where the maximum margin between different classes is learned, in One-Class SVM the boundary of the training data, i.e. Trojan free data, is learned. The decision function $f(\mathbf{x})$ assigns 1 in the region of the training points (HT-clean points), and -1 everywhere else (HT-contaminated points).

To introduce One-Class SVM we consider the training data consisting of n observations

$$\Omega = \{\mathbf{x}_i, i = 1, 2, 3, ..., n\}, \quad n \in \mathbb{N}$$

where $\mathbf{x}_i$ is the i^{th} observation (data-point) and it is assumed that Ω is mapped into a higher-dimensional feature space Ψ through a feature mapping Φ

$$\Phi : \Omega \to \Psi. \tag{4.1}$$

The mapping (4.1) is assumed such that the images of dot products $(\mathbf{x}_i \cdot \mathbf{x}_j) \in \Omega$ in the new feature space $(\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)) \in \Psi$ can be calculated via a kernel function [159]

$$k(\mathbf{x}_i, \mathbf{x}_j) = (\Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)). \tag{4.2}$$

To find a hyperplane in the new feature space which separates the data and the origin with a maximum margin the following minimising problem is formulated

$$\min_{\mathbf{w} \in \Psi, \boldsymbol{\xi} \in R^n, \rho \in R} \frac{1}{2}\|\mathbf{w}\| + \frac{1}{\nu n}\sum_i \xi_i - \rho, \tag{4.3}$$

with the following constraints

$$(\mathbf{w} \cdot \Phi(\mathbf{x}_i)) \geq \rho - \xi_i, \quad \xi_i \geq 0, \;\; i = 1, 2, ..., n. \tag{4.4}$$

In the quadratic programming problem Eqs.(4.3)-(4.4), which is called the primary problem, $\mathbf{w}$ and ρ are the parameters needed to be found, where $\mathbf{w}$ is the normal vector to the separating hyperplane, ρ is the margin between the mapped data and the origin, ξ_i are slack variables penalising the misclassifications. The regularization parameter $\nu \in (0, 1]$ is a user-defined parameter, which shows the fraction of outliers that should be allowed and n is the number of training points.

If the data in the input space is not linearly separable, which is often the case in one-class data, the so-called kernel trick is employed, which maps the inseparable initial data onto a separable data in the higher dimensional kernel space. This can be done by

writing the primary problem in the following equivalent form, called the dual problem [160]

$$\min_{\boldsymbol{\alpha}\in R^n} \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j k(\mathbf{x}_i,\mathbf{x}_j), \quad 0\le\alpha_i\le\frac{1}{\nu n}, \ \sum_i \alpha_i = 1, \tag{4.5}$$

where the kernel $k(\mathbf{x}_i,\mathbf{x}_j)$ is defined by the dot product Eq. (4.2), indices i and j range over $1,...,n$. The dual problem Eq. (4.5) does not require computation of the explicit map Φ in the objective function. It is sufficient to define the kernel instead. One of the most efficient kernels Eq. (4.2) for the dual problem Eq. (4.5), which will be used further is the radial basis function (RBF)

$$k(\mathbf{x},\mathbf{x}_i) = \mathrm{e}^{-\gamma\|\mathbf{x}-\mathbf{x}_i\|^2}, \tag{4.6}$$

where γ is the bandwidth parameter.

Further, the inequality constraints Eq. (4.4) in the primal formulation Eq. (4.3) (which are as many as the number of samples) are simplified to an equality and bound constraint in the dual problem Eq. (4.5). The algorithm finds the optimal values of α_i and only the solutions $\{\mathbf{x}_i\}$ with $\alpha_i > 0$, called support vectors, are taken in the final decision function.

After the minimization problem Eq. (4.5) is solved the required value of ρ can be found from the following formula

$$\rho = \sum_i \alpha_i k(\mathbf{x}_i,\mathbf{x}_j), \tag{4.7}$$

where $\mathbf{x}_j$ is any support vector. The new incoming data can be classified using the following decision function:

$$f(\mathbf{x}) = \mathrm{sgn}\left(\sum_i \alpha_i k(\mathbf{x}_i,\mathbf{x}) - \rho\right), \tag{4.8}$$

where $\mathbf{x}_i$ are the support vectors. The decision function is assigned 1 for data predicted as HT-clean, and -1 for HT-contaminated data.

4.3.3 Local Outlier Factor

Another classic anomaly detection method is Local Outlier Factor (LOF) [161]. The method estimates the local density of a given sample and the deflection from the density

of its neighbours. The local density is estimated by measuring distances of k nearest neighbours. Then samples are identified with considerably lower density and qualified as outliers.

The LOF approach is based on the concept of k-distance, $\|\mathbf{x} - \mathbf{x}^{(k)}\|$, which is the distance of a point $\mathbf{x}$ from its k^{th} nearest neighbours, and k-neighbours, $N_k(\mathbf{x})$, which are points in and on the circle of the radius k-distance.

Reachability distance (RD) from $\mathbf{x}$ to $\mathbf{x}'$ is

$$RD_k(\mathbf{x}, \mathbf{x}') = \max(\|\mathbf{x} - \mathbf{x}^{(k)}\|, \|\mathbf{x} - \mathbf{x}'\|), \tag{4.9}$$

where $\mathbf{x}^k$ is the k^{th} nearest neighbour of $\mathbf{x}$ and $\mathbf{x}' \in N_k(\mathbf{x})$. RD is used to define another key concept, the local reachability density (LRD) of $\mathbf{x}$:

$$\text{LRD}_k(\mathbf{x}) = \frac{1}{\frac{1}{\|N_k(\mathbf{x})\|} \sum_{\mathbf{x}' \in N_k(\mathbf{x})} RD_k(\mathbf{x}, \mathbf{x}')}. \tag{4.10}$$

It follows from Eq. (4.10) that greater LRD (i.e. neighbours far from the point $\mathbf{x}$) corresponds to lower density of points around a particular point. The algorithm uses the definition of the local outlier factor (LOF) of $\mathbf{x}$, which is the ratio of the average LRD of the k-neighbours to the LRD of $\mathbf{x}$:

$$\text{LOF}_k(\mathbf{x}) = \frac{\frac{1}{\|N_k(\mathbf{x})\|} \sum_{\mathbf{x}' \in N_k(\mathbf{x})} \text{LRD}_k(\mathbf{x}')}{\text{LRD}_k(\mathbf{x})}. \tag{4.11}$$

The aim of the LOF method is to determine an outlier by calculating the local outlier factor Eq. (4.11) for each data-point in the dataset [162]. Larger values of the LOF correspond to fewer data-points around the object. If for any point the LOF is less than 1, the density of the point is higher compared to that of its neighbours indicating that the point is HT-clean. If the LOF is greater than 1, then the point has less density compared to the density of its neighbours. In this case the point is possibly an HT-contaminated outlier.

4.3.4 Algorithm Complexities

The complexity of an algorithm is estimated by its time complexity, which assesses how long the algorithm takes to execute the task, and its spatial complexity, which assesses the amount of memory needed for the operation of the algorithm.

If an algorithm does not depend on the number of data N it processes then the algorithm is considered to have constant complexity which is denoted by $O(1)$. If the algorithm dependents on N, then the complexity is dependant on line code in the algorithm and can be estimated as $O(n), O(n^2), O(\log n)$ etc.

4.3.4.1 Complexity of One-Class Support Vector Machine

4.3.4.1.1 Time Complexity: There are two complexities to assess: training-time and run-time. Minimum training time complexity for SVM is estimated as $O(n^2)$, where n is the number of training data-points. The training process returns n_{SV} support vectors, which is later used for classification.

For a kernelised SVM with an RBF kernel Eq. (4.6), n_{SV} kernel computations are needed to classify a new point $\mathbf{x}$ in (4.8). Assuming for each support vector $\mathbf{x}_i$ the kernel Eq. (4.6) can be computed with $O(d)$ time complexity, where d is the dimension of the support vectors, the overall run-time complexity of the classifier with an RBF kernel is $O(n_{SV} \times d)$ for a single data-point under query. Although tuning of the γ parameter can have an impact on run-time as well since its computation has a quadratic complexity, One-Class SVM achieves good performance without significant tuning [163].

4.3.4.1.2 Spatial Complexity: The classification ready model, needs to know the coordinates of the support vectors, therefore all n_{SV} support vectors should be stored in memory. The dimensionality of the support vectors d matches that of the data for classification. Therefore the memory complexity for online prediction will be $O(n_{SV} \times d)$.

For a very large dataset, the training time for SVM with non linear kernel classifiers can be quite high. The complexity can be reduced by using a linear kernel $K(\mathbf{x}, \mathbf{z}) = \mathbf{x} \cdot \mathbf{z}$. In this case the decision function is $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$, where $\mathbf{w} = \sum_{l=1}^{m} \alpha_l y_l \mathbf{x}_l$ and classifying requires only $O(n)$ operations and $O(n)$ memory. However, linear kernel SVMs generally

have poorer classification performance [164].

The time and space complexities for models with an RBF kernel can also be significantly reduced by a number of approximation algorithms. Tsang et al. developed an approximation algorithm for SVM training that has $O(n)$ time complexity and space complexity independent of n [165]. This is achieved by showing that the soft-margin one and two-class SVMs can be considered as minimum enclosing ball (MEB) problems and treated by an efficient approximate MEB algorithm. In particular, solving the soft-margin One-Class SVM problem is essentially the same as fitting the MEB with outliers. Results show that using the RBF kernel on a set of intrusion detection data containing about five million training patterns takes only 1.4 seconds on a 3.2 GHz Pentium–4 PC [165].

The training time complexity of SVM can also be improved by employing the second-order polynomial approximation of the RBF kernel [166]. The classification performance can be close to that obtained with the exact RBF kernel, with complexity $O(d, (d+3)/2)$ which can be a significant improvement if the number of dimensions d is low compared to the number of support vectors in a model. Proper investigation is normally needed of the trade-off between performance and classification.

4.3.4.2 Complexity of Local Outlier Factor

Nearest-neighbour-based algorithms such as LOF have quadratic computational complexity since they have to calculate the distances of all data-points. Thus training time complexity of LOF algorithm is $O(n^2)$. Run-time and space complexities of LOF are estimated as $O(nd)$.

4.4 Proposed Methodology

Every electronic device, when performing a certain group of tasks, executes a unique set of commands in a predefined order. When an IC device (microcontroller, FPGA, ASIC) executes a specific command, the internals of the IC switch through a number of states. For every low-level command, a certain number of transistor switches occur in a strictly defined pattern. On the macro scale, this creates a specific power consumption pattern for

the IC for every high-level command [106]. Since a PCB is composed of a group of ICs performing their tasks, the same is true for a PCB. The proposed methodology identifies the underlying power consumption pattern for the device under test and feeds the data to a machine learning (ML) algorithm to learn the decision boundary[1] or some other classification criteria.

Every IC on the PCB has its specific set of tasks required for the PCB to operate as a whole. The exact ICs and their tasks can vary depending on the device and its use case. The device can have many different operational modes. Given the sensitivity of the proposed technique to the IC activities during run-time, the models have to be calibrated to predefined tasks of the PCB. For the proposed methodology to work the device should not be reprogrammable, changing the predesigned tasks. In other words, it is assumed that either through theoretical estimations or empiric means it is possible to collect an inclusive power consumption data from the golden model PCB, reflecting all of its legitimate operations. However, if it is not possible to build an inclusive dataset of PCB power consumption for all operational modes, e.g. due to their multitude, then further investigation will be required to adapt the existing data with further assumptions, to train the machine learning model.

As proposed in [141], every component on the PCB should be fitted with a dedicated power sensor, if they already don't have it built in. On every iteration, these sensors report their readings back to the monitoring block (MB) on the PCB, where all the processing is carried out. New datapoints are parsed through the ML model to get binary predictions of either 1 or -1 for HT-clean and HT-contaminated scenarios respectively. The proposed methodology is summarised in Figure 4.3, organised into three main stages: data acquisition and preprocessing, model training, and run-time monitoring.

In the data acquisition and preprocessing stage, first the power consumption data was collected from the PCB prototype (i.e. golden model). This is an HT clean dataset, part of which is later *contaminated* by adding HT power consumption values to some of the datapoints. To do the contamination process, the HT's parameters, such as power consumption values and the length of activation time, as well as the randomly

[1]The decision boundary is a closed shape in N-dimensional space, which determines the outcome of the classification prediction.

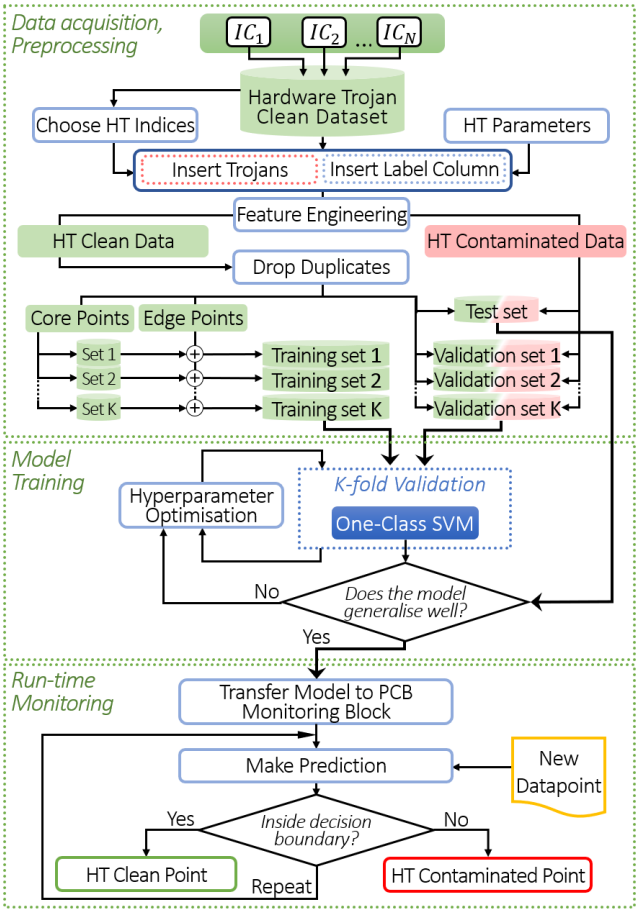


Figure 4.3: Process flow from data collection to real-time hardware Trojan monitoring on a PCB.

chosen locations in the dataset are chosen for every HT insertion instance. In this stage, HT-contaminated data has been added to the dataset using 100 distinct HT categories, later presented on page 86. Next, all the appropriate feature engineering is applied to the dataset, after which the HT-contaminated points are separated from HT-clean points, dropping all the duplicate points in the process. These points are organised into training, validation and testing datasets. While validation and testing datasets receive equal proportions of clean and contaminated points, the training dataset receives only HT-clean points. In order to improve the training process and the results, a technique is applied (see section 4.5.2) to remove redundant *core points*, greatly reducing the size of the training dataset, while making sure to retain all the *edge points*, which carry most of the boundary information required to train a good model. In such manner, K pairs of

training and validation datasets are created in the interest of arranging a K-fold validation like process while training the model in the next stage.

In the model training stage of the flow, first an appropriate ML algorithm is chosen to train on the data. The algorithm chosen in Figure 4.3 for demonstration is One-Class SVM. In this stage, the One-Class SVM model hyperparameters are tuned to the most optimal values for highest model accuracy. In the process of training the algorithm, K-fold validation like process has been arranged using the K pairs of training-validation datasets developed in the previous stage. This is done to increase confidence in adequate generalisation of the output model, rather than fitting it to the exact dataset applied during training. The resulting model is then tested on the test dataset, to confirm that it generalises well on previously unseen data, otherwise further hyperparameter optimisation is carried out.

Finally, **in the run-time monitoring stage**, the model is uploaded to the monitoring block (MB) on the PCB prototype to perform real-time monitoring of the device. When the PCB is turned on, the MB iteratively receives power readings from all linked power sensors. These readings are arranged into a single new datapoint in the form of a vector, which is then parsed through the One-Class SVM model to be monitored for an HT presence. If the new datapoint lies within the model's *decision boundary*, it is clean, otherwise it is classified as HT-contaminated and an HT is detected. This last step is performed iteratively for a continuous online monitoring of the PCB.

4.4.1 Notations

4.4.1.1 Data Point-Vector

On every iteration, the incoming data is stored in the MB. To organise the collected data in an orderly manner, it is grouped in $[1 \times N]$ dimensional data point-vectors:

$$\mathbf{X} = \{x_1, x_2, x_3, ..., x_N\}. \tag{4.12}$$

Since each vector component corresponds to a reading from one of the N power sensors (Figure 4.4), where every power sensor monitors an individual IC, the dimension N of vector $\mathbf{X}$ depends on the number of such sensors on the PCB.

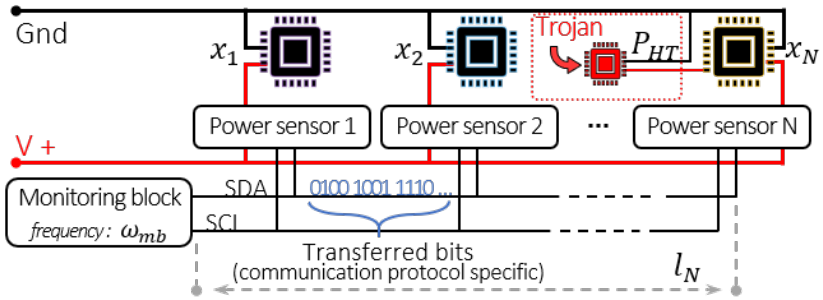


Figure 4.4: The proposed monitoring architecture and Trojan on an I/O of the N^{th} IC.

4.4.1.2 Classification Prediction

In the HT detection (classification) problem there are two possible labels. For HT-contaminated data-points, the label is assigned -1, while for HT-clean data-points the label is assigned 1. The label can be actual ($\mathbf{y}$) or predicted ($\hat{\mathbf{y}}$). The actual label $\mathbf{y}$ carries the ground truth information and is considered correct. The predicted label $\hat{\mathbf{y}}$, on the other hand, is the label given by the trained ML model in the prediction process and can potentially be erroneous. In supervised classification problem, the quality of the trained ML model is measured by the parity of the actual label $\mathbf{y}$ and predicted label $\hat{\mathbf{y}}$, with the best case being a complete match between the two.

4.4.1.3 Classification Metric

Precision and Recall rates are defined by the following formulae:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}, \tag{4.13}$$

where TP, FP, and FN are the numbers of true positive, false positive and false negative predictions respectively.

Depending on the end use case of a particular ML model, priority may be given to either high Precision or high Recall rate. In this work Precision and Recall have been treated with equal weights and the ML model has been optimised for the highest balanced F1-score. F1-score is an imbalanced-dataset aware classification metric and the closer it is to 100% the fewer misclassifications there are. The balanced F1-score can be calculated

using the following formula:

$$\text{F1-score} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{4.14}$$

4.4.1.4 Hardware Trojan Power Consumption

Depending on the chosen model, power consumption of the Trojan, denoted by P_{HT} (Figure 4.4), can be modelled using different probability density functions (uniform, normal, etc.), with their respective parameters. In this work, the HT power consumption has been modelled using a number of Gaussian normal probability density functions

$$f_i(P_{HT}) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left[\frac{-(P_{HT} - \bar{P}_{HTi})^2}{2\sigma_i^2}\right], \;\; i \in \mathbb{N} \tag{4.15}$$

where $\bar{P}_{HTi}$ is the mean and σ_i is the standard deviation of the i^{th} HT. However, it should be noted that there is no strong motivation for using this particular distribution for the HT's power consumption model. Choosing an alternative distribution, e.g. uniform, is not expected to have a significant impact on the results of proposed methodology.

4.4.1.5 Hardware Trojan Active Time

The HT active time span is denoted by T_{HT}. On every iteration the MB generates a data point-vector. The vector contains power consumption information gathered from all the power sensors, one of which can potentially bear an HT-contaminated IC. In order to measure the active time span of the HT, a sufficiently general time evaluation metric is necessary since the probing frequency of the monitoring setup will vary depending on the characteristics of different real-life PCB devices (e.g. working frequencies, number of ICs). Such an evaluation metric can be the number of consecutively recorded HT-contaminated data point-vectors i.e. HT-points. In other words, one HT-point is the time required for a single power consumption data point-vector to be registered by the MB, while the HT has been operating and consuming power. The relation of the HT active time to real time can be expressed via the number of HT-points using the time unit defined as

$$Unit\ HT\text{-}point = \alpha_1 \frac{N_{sensors}}{\omega_{MB}} + \alpha_2, \tag{4.16}$$

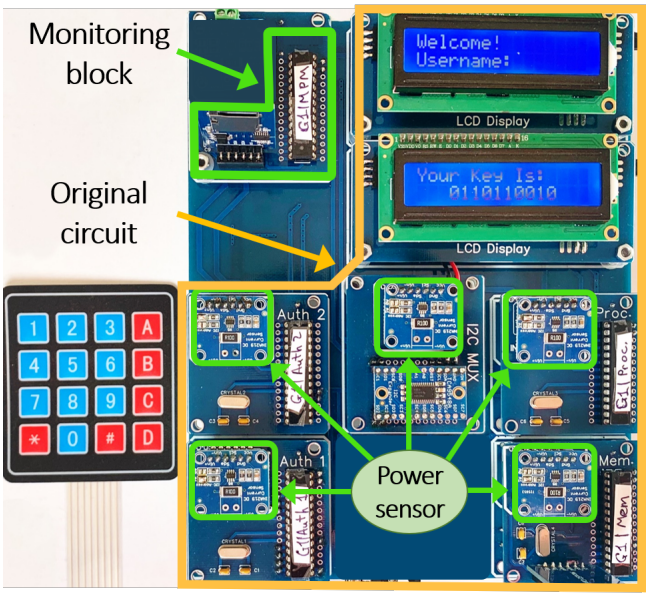


Figure 4.5: Printed circuit board prototype.

where coefficient α_1 depends on the operational speed of the communication protocol, $N_{sensors}$ is the number of sensors, ω_{MB} is the operating frequency of the monitoring block and α_2 depends on the characteristics of the PCB such as the layout and the length of data wires (e.g. l_N) between power sensors and the monitoring block (Figure 4.4).

4.5 Data Collection and Hardware Trojan Modelling

Prototype PCB in Figure 4.5 has been developed for real-life power consumption data collection. Subsequently, 100 categories of HT devices have been modelled and injected into the data for validation and testing of the trained ML model. Although it is possible that more than one IC can be infected with an HT at the same time, only single HT scenarios have been considered in this work.

4.5.1 The Prototype Device

4.5.1.1 The Original Circuit

The function of the original circuit (Figure 4.5, marked by orange border) is to store and display data from the built-in memory block, after a log-in occurrence. On

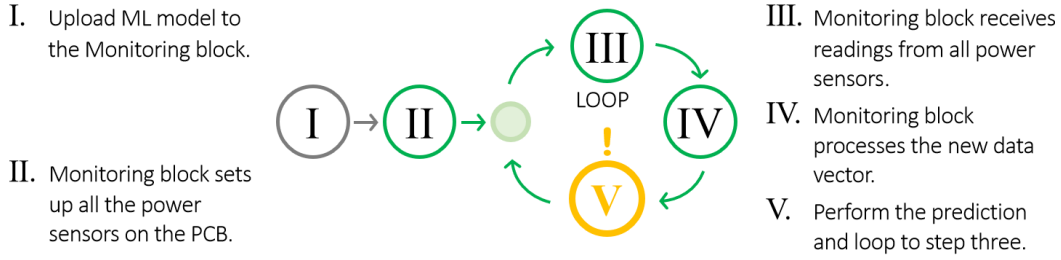


Figure 4.6: Monitoring block run-time power monitoring and Trojan prediction steps.

the prototype PCB there are five IC devices under power monitoring. A keyboard and a display, linked to an authentication block, are used to facilitate the log-in process. Upon a successful log-in event, an enable signal is generated by the authentication block. This signal is fed into a processing block. The enable signal triggers the processing block to fetch the secret data from a memory block. Here, the processing block uses Inter-Integrated Circuit (I^2C) communication protocol to request from the memory block. In turn, the memory block requests the data from a built in memory device through Serial Peripheral Interface (SPI). Once this secret information is passed to the processing block, it is presented on a second display. The communication with both displays is executed through I^2C protocol. Before long, the system automatically logs out, preparing for a new log-in cycle. While performing their tasks the power consumption of the legal ICs vary between $\sim 10\ mW$ and $\sim 50\ mW$.

4.5.1.2 The Monitoring Circuit

There are two main blocks in the power monitoring circuitry: (a) a monitoring block, and (b) five power sensors (one for every IC) (Figure 4.4, Figure 4.5). Inside the monitoring block an ATmega328P microcontroller has been used to perform the on-board data collection, processing and prediction tasks and to deliver the external communication functions. The microcontroller is linked to all power sensors via a multiplexer for communication using the I^2C protocol.

The action flow inside the monitoring block is illustrated in Figure 4.6. After the previously trained machine learning model has been uploaded onto the monitoring block (step I), the monitoring block sets up all the power sensors on the PCB (step II). Next, in step III the monitoring block receives power readings from all the sensors and stores

the data in a $[1 \times N]$ dimensional row vector. Then, in step IV any processing of the new point is performed (e.g. computation of moving averages). The resulting data point-vector is parsed through the ML model in step V. Finally, if the resulting prediction is $\hat{\mathbf{y}} = -1$, suggesting an HT-contaminated case, an HT intrusion alarm is raised, otherwise the microcontroller loops back to step III and begins the next iteration. Note that within one iteration the time-span between power readings from any two power sensors is assumed significantly smaller compared to the HT active time. Power sensing is executed through INA219 high-side current and power monitor chips. These chips have a built in 12 bit ADC and provide readings with 1 mW resolution.

4.5.2 Data Collection and Feature Extraction

An important aspect of this research is that the data used to train, validate and test the machine learning models is collected from a real-life device, rather than being computer generated. To collect the data, the monitoring block on the prototype PCB device has been fitted with a micro-SD memory card to store the power consumption data when there is no HT device mounted on the PCB. Every power consumption data-point is generated as an N dimensional $[1 \times N]$ row-vector $\mathbf{X}_i$. It is stored in rows and columns, where every row corresponds to one iteration of data collection and every column corresponds to a specific IC (Table 4.1). During the data collection stage over 3×10^6 data point-vectors were recorded, which accounted for all the working modes of the PCB device. These data-points are distributed in several clusters in $\mathbb{R}^N$, their layout depending on the power consumption patterns of the actual PCB device under investigation. In this work, the number of these clusters has been 16. The vast majority of the data-points are scattered in the near vicinity of the centre of the cluster they belong. These points can be referred to as core points. The points that are closer to the peripheries of the clusters, referred to as edge points, are far lower in number and hence there is a high likelihood that a random selection of a fraction of all the points for model training purposes will miss most, if not all, of the actual edge points. This unfair selection will result in a skewed and consequently wrong power consumption pattern illusion in the training dataset. Given the finite number of points in validation and testing datasets, the simulation outcome might be reasonable. However since the training dataset will not be fully representative of the actual power

consumption distribution pattern, ultimately the performance of the classification models in a real-life continuous run-time experiment will suffer, resulting in a significant drop of classification F1-score.

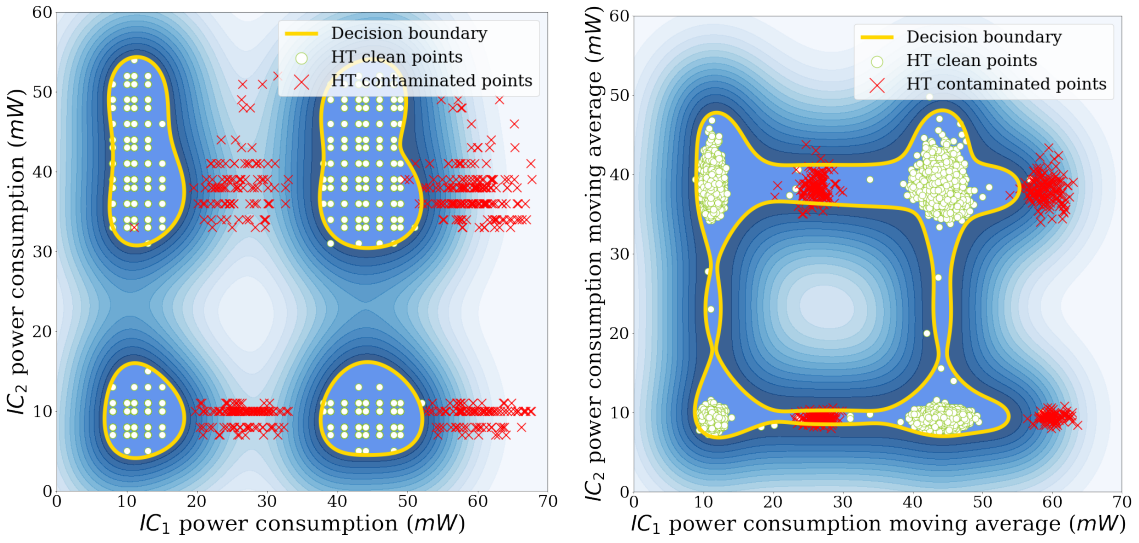
Table 4.1: Insertion of n-point Trojan occurrence on IC_3.

Index	**IC$_1$**	...	**IC$_3$**	...	**IC$_N$**	**y**
$\mathbf{X}_0$	$x_1^{(0)}$	...	$x_3^{(0)}$	...	$x_N^{(0)}$	1
...	...	...	...	...	...	...
$\mathbf{X}_i$	$x_1^{(i)}$	...	$x_3^{(i)}$	...	$x_N^{(i)}$	1
$\mathbf{X}_{i+1}$	$x_1^{(i+1)}$	...	$x_3^{(i+1)}+P_{HT1}$	...	$x_N^{(i+1)}$	-1
$\mathbf{X}_{i+2}$	$x_1^{(i+2)}$	...	$x_3^{(i+2)}+P_{HT2}$	...	$x_N^{(i+2)}$	-1
...	...	...	...	...	...	...
$\mathbf{X}_{i+n}$	$x_1^{(i+n)}$	...	$x_3^{(i+n)}+P_{HTn}$	...	$x_N^{(i+n)}$	-1
$\mathbf{X}_{i+n+1}$	$x_1^{(i+n+1)}$	...	$x_3^{(i+n+1)}$	...	$x_N^{(i+n+1)}$	1
...	...	...	...	...	...	...

To solve this problem, an algorithm called *Density-Based Spatial Clustering of Applications with Noise* (DBSCAN) has been employed to identify and isolate edge and noise points. The noise points have later been removed, while, more importantly, all the edge points have been included in the training dataset, along with randomly selected core points to fill in the core of the clusters. Thus, the training dataset consists of all the edge points, some of the core and almost none of the noise points (*stage "Data acquisition, preprocessing"* in Fig 4.3). This technique helps speed up the model training process by allowing for smaller training datasets, as well as making the process more effective at finding a better boundary between expected HT-clean and deviant HT-contaminated data-points (Figure 4.7a).

4.5.3 Hardware Trojan Modelling and Insertion

The insertion of hardware Trojans on the PCB is demonstrated by imitating the effects it would have caused, i.e. increased power consumption due to its operations. The HT model has been developed to replicate an extra triggerable component on the PCB, powered



(a) Unmodified readings from IC_1 and IC_2

(b) Averaged readings from IC_1 and IC_2

Figure 4.7: Two dimensional cross sections of the four dimensional dataset and the One-Class SVM decision boundary.

from the I/O pins of a legitimate IC (Fig 4.2b). The HT free data has been superimposed with the modelled HT's power consumption at random locations, thus effectively replicating a realistic model of an implanted triggerable HT. In this model the power consumption of an active HT, P_{HT}, follows a Gaussian normal probability density function (4.15) and hence the HT model has three parameters, HT's mean power consumption $\bar{P}_{HT}$, the standard deviation σ and the active time span T_{HT}. The model generates T_{HT} number of HT device power consumption values, which are normally distributed around $\bar{P}_{HT}$ with a standard deviation set to σ. Note that the duration of the HT's active time is measured in HT-points (subsection 4.4.1.5).

The HT with a triggering mechanism has a state of active payload delivery after being triggered (state 1) and an idle state (state 2). The power consumption of the Trojan device in the first state is normally distributed around $\bar{P}_{HT}$, with a standard deviation of σ. In state 2, the power consumption is assumed to be $\sim 0\ mW$, i.e. too low to be detected by the sensors, mimicking HT's power down or idle modes. Note that the proposed method is invariant of the payload of the HT, as long as it consumes extra power. An abstraction of a simulated HT's power consumption data group for a single activation occurrence can

be presented as follows

$$\mathbf{P_{HT}} = \{P_{HT1}, P_{HT2}, P_{HT3}, ..., P_{HTn}\}, \tag{4.17}$$

where n is the number of HT-points T_{HT}.

Several generated HT data groups similar to Eq. (4.17) have been randomly added to the validation and test datasets in such a way that the ratio of the HT-contaminated vectors to HT-clean vectors is 1:1 (i.e. 50% HT-clean and 50% HT-contaminated). While adding the HTs, a random choice has been made of both the columns (i.e. ICs on the PCB) and the rows (i.e. HT activation points in time), while assuring that no overlapping of rows takes place. Table 4.1 illustrates how the HT-clean data-frame is modified to become an HT-contaminated data-frame. In this particular example an n-point long HT occurrence has been added in column IC_3, after instance X_i point in time, i.e. an HT activation occurrence on IC number 3 at time *i+1*. In Table 4.1, it can be seen how the power consumption of the HT device (in red) is added to that of the legitimate IC. In this work, 100 categories of Trojan devices have been modelled and injected into the original HT-clean data collected from the prototype PCB. Table 4.2 provides ranges for the HT's mean power consumption $\bar{P}_{HT} \in \{[5:50]$, step $= 5\ mW\}$ and active-time HT-points $T_{HT} \in \{[10:100]$, step $= 10\ point\}$ (described in section 4.4). The hundred HT categories were generated by taking combinations of the provided parameter values, while the standard deviation is $\sigma = 1$ for all.

Table 4.2: Parameters chosen for HT models.

	Mean power, $\bar{P}_{HT}$	Active time, T_{HT}
HT Categories	5 mW to 50 mW step size 5 mW	10 *point* to 100 *point* step size 10 *point*

4.5.4 Feature Engineering

In an effort to reduce the noise in the sensor readings and improve classification outcomes, feature engineering has been applied. For every column in the dataset, a new column has been appended, which was populated with the moving averages of the original values. This way, using the moving average, the power reading row-vectors X_i (Table 4.1)

are filtered by taking the column-wise average with their respective previous $M-1$ values,

$$X_i = \frac{1}{M} \sum_{j=0}^{M-1} X_{i-j}, \quad \text{for} \quad \forall\, i \in \mathbb{N} \quad \text{s.t.} \quad i \geq M, \tag{4.18}$$

where M is the number of averaged points. The optimal value for parameter M can be estimated empirically. In this work M has been set to 5. After applying feature engineering the dataset matrix with N columns becomes one with $2N$ columns, changing the dimension of every data point-vector from $[1 \times N]$ to $[1 \times 2N]$.

4.6 Model Training and Simulation Results

In order to create reliable models, supervised ML algorithms should use sufficiently large datasets, which effectively capture the underlying data distribution patterns. These trained models are later utilised for either regression or classification problems [167]. The end-to-end process consists of several steps. Generally, the first step would be initial data manipulation including data cleaning and feature selection where the relevant information-bearing features are extracted from the raw data. Next, feature engineering may be applied where the data is manipulated in some way, which increases its predictive power. The resulting sample data is divided into training, validation and test datasets and stored for later procedures. The second step is selecting and executing the appropriate ML algorithms to obtain the end models, fitted on the training dataset and optimised on the validation dataset. Here, among other procedures, grid-search optimisation of algorithm hyperparameters is carried out in conjunction with K-fold validation to determine the best model. In the final evaluation stage, the performance of this model is assessed on the test dataset. The optimal model is chosen, which will be used to make run-time predictions for new data on the PCB.

In the problem set out in this work the training data consists of only one class, the HT-clean class. Having collected all the necessary data, two machine learning classification algorithms have been trained: One-Class Support Vector Machine and Local Outlier Factor. Although the training parameters vary for the algorithms, the task at hand is the same for both of them - find the best possible decision function, which will maximise the

number of correctly predicted HT-clean and HT-contaminated data-points, while avoiding any unnecessary overfitting to the training data. This way the model should be able to best predict if a new incoming data-point belongs to the HT-clean class or has a significant deviation and so can be safely classified as an anomaly. All the anomaly points are considered as HT-contaminated.

As an example consider a PCB with only two ICs and their respective power sensors. This sets the dimensions of the data point-vector at $N = 2$. Then, with the feature engineering applied, the final dimension of the data point-vector becomes $2 \cdot N = 4$. Such data point-vectors reside in a four dimensional space, therefore the decision boundary is a four dimensional surface. For visualisation purposes two double-feature cross sections of the four dimensional shape are shown in Figure 4.7. The two feature axes in Figure 4.7a represent power consumption readings received from the two power sensors on IC_1 and IC_2. The axes in Figure 4.7b, on the other hand, are the feature engineering generated moving averages of the same power readings. Shown in white dots are the HT-clean data point-vectors, while the red crosses represent the HT-contaminated data point-vectors. The golden bands illustrated in the diagrams are the 2 dimensional cross sections of the determined 4 dimensional decision boundary (a hypersphere in the kernel space). Any new point enclosed by the golden band will be classified as an HT-clean vector, while the ones outside will be classified as HT-contaminated vectors. The HT-to-IC power consumption ratio has been considered under variability of the power consumption points of legal ICs. The more compact the power consumption clusters of legal ICs are (Figure 4.7a), the lower-power HTs will become detectable since they will not be concealed within the boundaries of legal power consumption. Shown in Figure 4.7a, notice how some of the HT affected red points remain within the bold yellow decision boundaries because of the width of the respective cluster.

One-Class SVM algorithm has been chosen to determine the decision boundary in Figure 4.7. Once the best model has been trained the decision function is uploaded onto the monitoring block on the prototype device to perform run-time monitoring of the PCB. Pseudo-code for the online monitoring steps is provided in Algorithm 2.

Algorithm 2 One-Class SVM run-time monitoring.

Input: *M, N, SVM*$(x) \rightarrow$ *Decision function*
Initialise: $k = 0$
loop $k = k + 1$
    for *i=1,2,3,..., N* ***do*** ▷ *Collect sensor data*
        $\boldsymbol{X}[k][i] = Sensor^{(i)}$
    end for
    for *j=1,2,3,..., N* ***do*** ▷ *Feature engineering*
        $\boldsymbol{X}[k][N+j] = \frac{\boldsymbol{X}[k][j] + ... + \boldsymbol{X}[k-M][j]}{M}$
    end for
    $\hat{y} = SVM(\boldsymbol{X}[k])$ ▷ *Make prediction*
    return $\hat{y}$
end loop

4.6.1 One-Class Support Vector Machine

Parameter Selection

The bandwidth parameter γ in Eq. (4.6) and the regularization parameter ν in (4.3) and (4.5) are the main parameters in One-Class SVM and play key roles in the quality of the resulting model [168]. Smaller ν values leave fewer training samples on the origin side of the hyperplane. If ν approaches to 0, the upper bounds on the Lagrange multipliers α_i tend to infinity in Eq. (4.5). The penalty term $\sum \xi_i$ in the objective function Eq. (4.3) vanishes, which means that the hyperplane found in the feature space separates almost all the training data from the origin. If ν approaches to 1, the algorithm leaves more data-points on the origin side of the hyperplane, when $\nu = 1$ almost all the data-points are classified as anomalies.

Training

To reduce the degree of uncertainty in the results, ten-fold validation has been carried out, while running a greedy grid-search over the hyperparameters ν and γ. The moving average hyperparameter M has been set to five during the feature engineering stage. Next, the resulting One-Class SVM model has been tested on the test set to check if it generalises well on previously unseen data. While the training dataset contains HT-clean class only, the validation and test datasets are composed of 50% HT-clean and 50% HT-contaminated points, including all 100 HT categories from Table 4.2 on page 86.

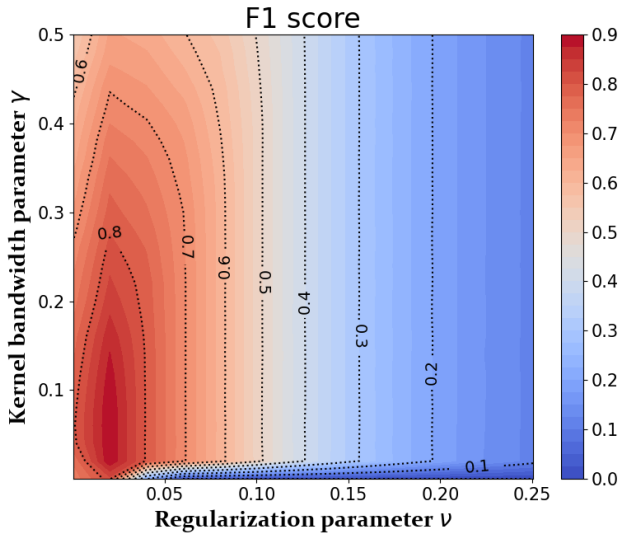


Figure 4.8: One-Class SVM classification F1-score contour map.

Results

The contour map shown in Figure 4.8 illustrates F1-score dependency of One-Class SVM regularization hyperparameter ν and kernel bandwidth γ. It can be seen in the figure that F1-score of the trained One-Class SVM model peaks in the lower-left-corner of the (ν, γ) plane. In fact a greedy grid-search for the best hyperparameters has returned $\{\nu = 0.0015, \gamma = 0.016\}$. Using this $\nu - \gamma$ pair the ML model reaches prediction F1-scores 0.993, 0.968 and 0.968 on training, validation and testing datasets respectively.

The classification results per HT category are shown in Figure 4.9. It can be seen that, indeed, HTs with lower power consumption are harder to detect. Comparing HT categories, when $\bar{P}_{HT} \geq 40\ mW$ the F1-score reaches above 99%, while when $\bar{P}_{HT} = 5\ mW$ the detection rates peak at F1-score = 82.6%. Trojan activation times, on the other hand, do not seem to have such a dramatic impact on detectability as HT power consumption does. It can also be seen that there is a drop in F1-scores around $\bar{P}_{HT} = 30\ mW$. This is a direct result of the blind zone effect described in Figure 4.2a, when the HT's added power consumption shifts the red curve to the right by exactly the amount necessary to overlay the left red peak with the right green peak. The graph presented in Figure 4.10 helps visualise the simulation results, while also showing the Precision and Recall rates and the average of all F1-scores for $T_{HT} = 10$ time points.

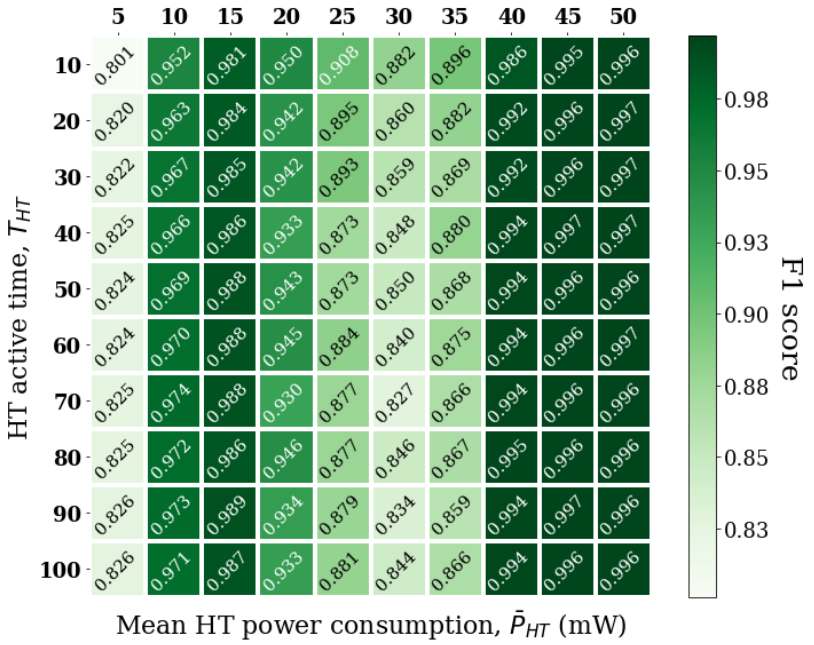


Figure 4.9: One-Class SVM classification results for all Trojans.

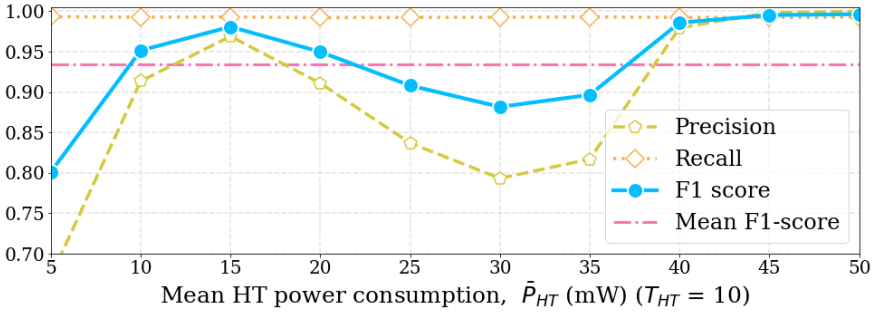


Figure 4.10: One-Class SVM classification results for all Trojan power consumption values ($T_{HT} = 10$).

4.6.2 Local Outlier Factor

This algorithm, defined in Eq. (4.11), takes its name after the anomaly score of each sample. This is a measure of the local deviation of density of a single data-point in comparison with that of its neighbours. The locality aspect plays a role in that the anomaly score depends on the degree of isolation of the given data-point with respect to its own neighbourhood. The distance from k-nearest neighbours Eq. (4.9) is used to estimate the local density. A comparison of the local density of a sample and the local densities of its neighbours can reveal the HT-contaminated points, which would have much lower density.

Parameter Selection

The main hyperparameters in LOF that control the quality of the classification model are the number of neighbours k and the contamination factor c. The parameter k is used to calculate the value of *k-distance*, required for the LOF model to operate. The contamination factor c is the ratio of the number of deviant (anomalous) points in the data and that of the normally expected (HT-clean) points. The parameter c is used to control the precision of training. It is worth noting that, if k is set larger than the number of available points in the dataset, all the available points will be used in the calculation.

Training

Similar to the previous algorithm, LOF too has been trained using 10-fold validation with the same datasets as used with One-Class SVM. Here too, the 100 HT parameter combinations have been chosen from Table 4.2 on page 86. The moving average hyperparameter M again has been set to five during feature engineering. The model has been optimised for the highest F1-score by running a greedy grid-search over the hyperparameters k and c.

Results

The classification results for LOF are shown in Figure 4.11. Comparing values from the first columns of Figure 4.9 and Figure 4.11, it can be seen that in the lower spectrum of HT's power consumption LOF algorithm outperforms One-Class SVM on average by 8.5% when $\bar{P}_{HT} = 5\ mW$. On the other hand, for $\bar{P}_{HT} \geq 5\ mW$ One-Class SVM proves to be a more robust algorithm for HT detection, eventually reaching F1-scores above 99.7% for $\bar{P}_{HT} = 50\ mW$, while LOF reached its maximum at 97.1%. Regarding HT active time length, as before T_{HT} has a more subtle effect on the detectability. That being said, when $\bar{P}_{HT} = 5\ mW$, F1-score for $T_{HT} = 100$ is 3.1% higher than that for $T_{HT} = 10$. In Figure 4.11 it can also be seen that for LOF too the blind zone effect results in a drop in F1-scores around $\bar{P}_{HT} = 30\ mW$. The graph in Figure 4.12 ($T_{HT} = 10$) clearly shows this drop in F1-scores when $20\ mW \leq \bar{P}_{HT} \leq 40\ mW$, while also providing values of the Precision, Recall and average of all F1-scores.

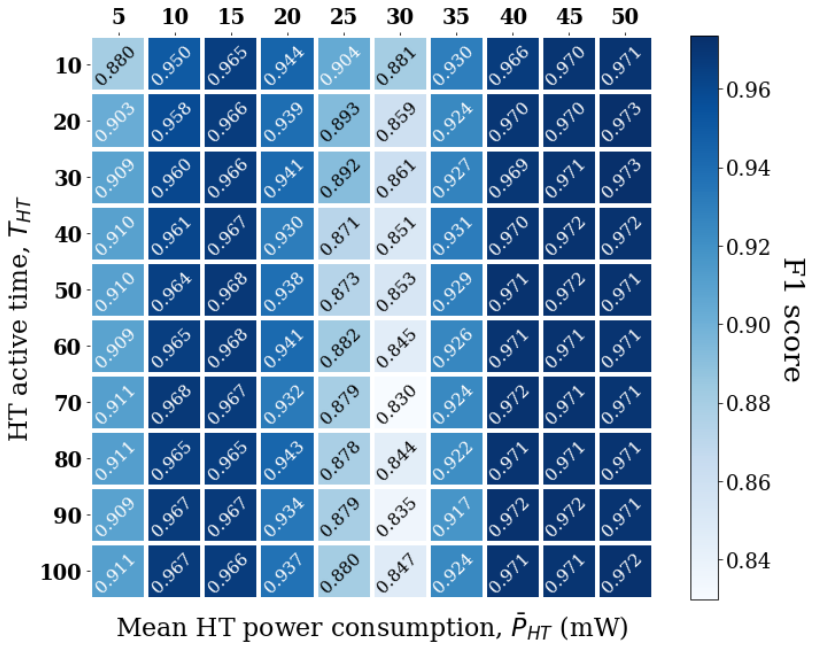


Figure 4.11: LOF classification results for all Trojans.

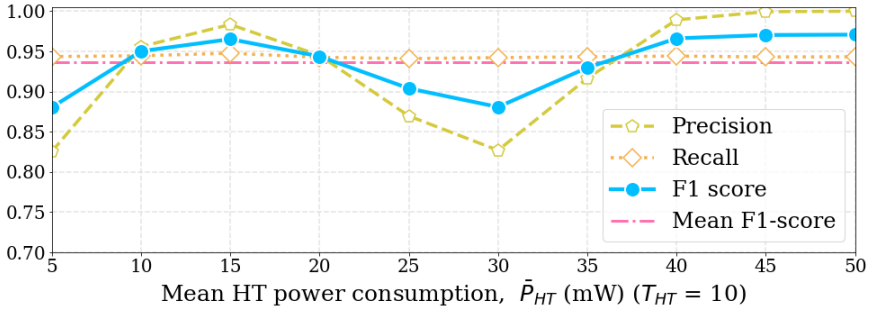


Figure 4.12: LOF classification results for all Trojan power consumption values ($T_{HT} = 10$).

4.7 Experimental Results

To verify the effectiveness of the proposed methodology and validate the simulation results, several experiments have been performed on the PCB prototype shown in Figure 4.5, using One-Class SVM as the classification algorithm. The reason why One-Class SVM is chosen for the experiment is that it generally outperformed LOF during the simulations. To further justify the choice, One-Class SVM is an eager-learner algorithm, requiring less computational load at prediction phase. In contrast, LOF is a lazy-learner [169] and

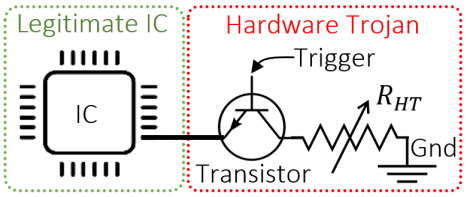


Figure 4.13: Schematic diagram of the deployed HT.

defers the majority of computation to prediction phase. To be feasible for production, an on-board real-time monitoring ML model should have small footprint on computing and memory resources.

First, the trained One-Class SVM model has been transferred to the monitoring block on the prototype PCB. That includes coding the decision function (4.8) and uploading coordinates of all the support vectors $\mathbf{x}_i$, the α_i coefficients for every support vector, as well as the free term ρ in the decision function. Next, a triggerable HT device has been added to the PCB using a bipolar junction transistor (BJT) and a range of resistors with varying resistances (Figure 4.13). The trigger sequences have been recorded in sync with the power consumption data on the PCB for later model evaluation analysis. The HT resistance values have been carefully chosen so that their power consumptions P_{HT} lie within the ranges 5 mW–10 mW, 10 mW–15 mW or 15 mW–20 mW. The duration of HT's active times T_{HT} are one of {20, 50, 100}. Shown in Figure 4.14 are the prediction F1-scores for all nine combinations of P_{HT} and T_{HT}. The figure compares One-Class SVM results from the experiment with those from One-Class SVM and LOF from simulations.

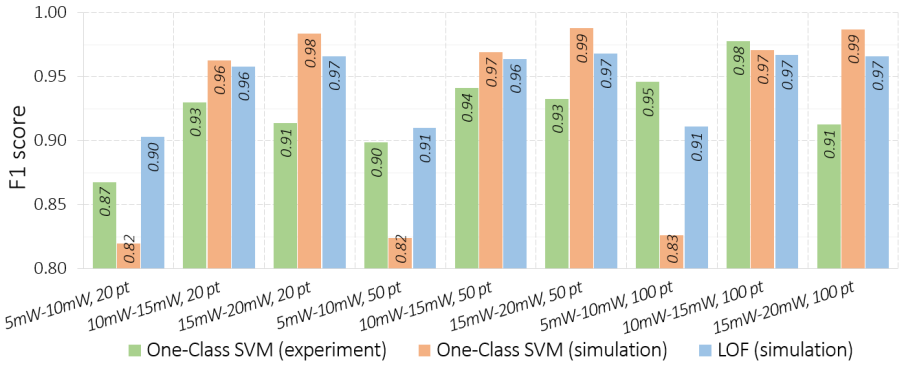


Figure 4.14: Comparison of experiments and simulations.

Memory usage

Given that the dataset has dimensionality of 10, using 16 bit unsigned integer encoding, a One-Class SVM model with 1258 support vectors will require memory capacity of about 27 KB. To show the memory capacity requirement calculation consider for example a model with 500 support vectors (500 $\times$ 10 integers), 500 support vector coefficients (500 integers) and a single free term ρ will need to store in memory a total of about 10 KBs of parameters.

$$\text{Memory} = \frac{(500 \cdot 10 + 500 + 1) \cdot 16}{8} \approx 10 \ KB \tag{4.19}$$

4.8 Discussion

The detectability of HTs can be affected by issues overlooked in this work. Comparing Figure 4.10 and Figure 4.12, it can be seen that the two algorithms have different behaviours at the two ends of the considered power consumption spectrum. While One-Class SVM algorithm has better performance at the upper end of the spectrum (*F1-score = 0.997 at 50* mW), LOF algorithm shows superior results at the lower end (*F1-score = 0.88 at 5* mW). For both algorithms, however, the HT's mean power consumption of *5* mW (*standard deviation = 1* mW) is the region where the performance begins to decay. One reason for such drop in performance is that the resolution of the power sensors used in this research is at *1* mW, which is 20% of the HT power consumption. Improving the resolution of the sensors on the prototype will reduce the noise-to-signal ratio, improving data quality and thus the results.

In contrast to the prototype used in this study, PCBs may have a larger setup with more ICs on board. Hence, a discussion of possible effects on the proposed methodology is due. From the methodology standpoint, performance degradation is expected to happen for two main reasons, including slower data collection iterations according to Eq. (4.16) given the increased number of power sensors, and slower computation due to higher data dimensionality. However, this can be alleviated by deploying a more powerful microcontroller in the monitoring block. From the ML algorithm standpoint,

a phenomenon commonly referred to as the curse of dimensionality could take effect when increasing the number of dimensions in data, i.e. the number of ICs on the PCB. ML algorithms based on distance measures tend to fail when the number of dimensions in the data is very high. In the Euclidean space, this happens due to noise-induced reduction of the ratio between distances of datapoints belonging to different classes and points from the same class, i.e. numerically all points appear closer in the high dimensional space. This effect results in a declined performance in algorithms such as One-Class SVM. Note, it typically takes place after several hundred dimensions, which should be plenty for the PCB application scenario. Furthermore, this problem could in principle be mitigated with dimensionality reduction techniques, e.g. Principal Component Analysis.

In some cases, a single PCB can have several power distribution networks (PDNs) with different electrical characteristics. Such instances should not affect the conclusion drawn from this work. This is because the power sensors are measuring the power consumption on the end nodes of the PDNs, regardless of the specific characteristics such as voltage value. From the methodology's standpoint these could be regarded as separate entities. The model does not require awareness of the PDN layout of the PCB and, therefore, it can be trained for such scenarios.

Ultimately, given the wide spectrum of possible Trojan attacks, a comprehensive solution should comprise of a combination of methodologies targeting a subgroup of all possible Trojans. In this chapter, the group of active Trojan implant ICs has been addressed, which consume a reasonable amount of power on the PCB. The group of ultra-low power passive Trojans, e.g. resistors and capacitors, should be addressed by other techniques such as visual inspection. For example, in a recently published paper [170] a visual inspection technique has been proposed using optical images to highlight the sections on PCB with a high likelihood of harbouring a surface mount Trojan component.

4.9 Concluding Remarks

In this chapter a combination of power analysis with machine learning (ML) algorithms has been implemented with the purpose of detecting hardware Trojan (HT) components on a Printed Circuit Board (PCB). The monitoring circuit architecture proposed by Piliposyan et al. [141] is further developed with the introduction of ML methods to detect stealthier HTs powered from legitimate chips on a PCB.

Two ML algorithms have been applied to the power consumption data collected from a purpose built PCB prototype (Figure 4.5). Later this dataset has been injected with one hundred categories of HTs from Table 4.2 with power consumptions between 5 mW and 50 mW. The resulting dataset has been divided into three datasets (training, validation and testing) to train One-Class Support Vector Machine (SVM) and Local Outlier Factor (LOF) one class classification algorithms. Simulations for both algorithms returned classification results reaching F1-scores above 99.5% and 97% for One-Class SVM and LOF respectively, given the HT's mean power consumption is $\bar{P}_{HT} \geq 45\ mW$. The analyses also showed that the length of HT active time T_{HT}, as defined in section 4.4, does not have significant impact on detectability.

Further, the One-Class SVM model has been uploaded to the monitoring block of the PCB. The simulation results have been validated through real-life experiments carried out on the prototype PCB. Comparison between simulation and experimental results has been presented for nine hardware Trojan categories, their power consumption ranging from 5 mW to 20 mW. Further, the One-Class SVM model is low-cost and requires only 27 KB memory to operate.

The objectives set out in section 2.7 concerning the second contribution have been met in this chapter. Chapter 3 and Chapter 4 together have covered the whole circuitry of the PCB, leaving no blind spots for an adversary to install an active HT undetected. The next chapter proposes a different approach, visual inspection, which unlike power analysis does not require the addition of monitoring circuitry. An automated visual inspection method, based on a combination of conventional computer vision techniques and a deep neural network, aims to detect HTs implanted on the surface of a PCB.

Chapter 5

Computer Vision for Hardware Trojan Detection on a PCB

With advances in technology hardware Trojan (HT) attacks on printed circuit boards (PCB) are becoming more sophisticated and the need for more effective HT detection methods is becoming crucial. Automated visual inspection (AVI) is one of the most promising solutions in detecting malicious implants on a PCB. It is non-destructive, effective in testing PCBs on an industrial scale, demands minimum human involvement, and can potentially identify malicious inclusions and modifications on PCBs at all stages of production and thereafter. In recent years, machine learning algorithms in particular in image recognition, localization, segmentation, and other areas, have been successfully applied, significantly improving the effectiveness of AVI methodologies.

In this chapter, an AVI methodology is proposed for detecting HTs on a PCB, using input data from a low-cost digital optical camera. It is based on a combination of conventional computer vision techniques and a dual tower Siamese Neural Network (SNN), modelled in a three stage pipeline. Further, a dataset of PCB images has been developed in a controlled environment of a photographic tent.

5.1 Introduction

Several printed circuit board (PCB) assurance techniques have already been suggested and evaluated over the past years. These include in-circuit testing, functional testing, Joint Test Action Group (JTAG) boundary scanning and bare-board testing. Each of these assurance methods have advantages and limitations, i.e. situations where they can be less effective [5]. With advances in technology, Trojan attacks become more sophisticated and

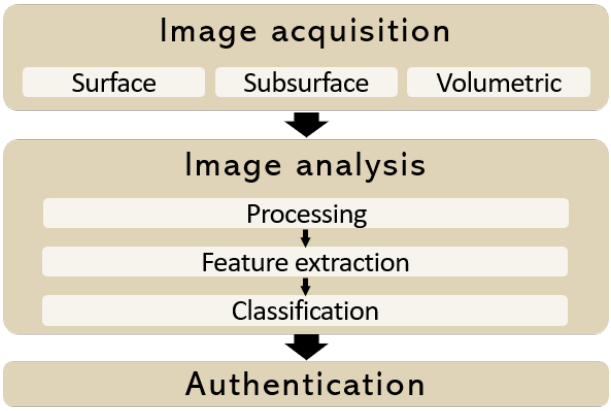


Figure 5.1: Overview of automated visual inspection process.

so there is a growing demand for more effective Trojan detection methods. Ideally, the most effective solution would require minimal human involvement, be non-destructive and be able to detect as many types of malicious modifications, inclusions and defects on a PCB as possible.

Automated visual inspection (AVI) has the potential to satisfy all these expectations. It requires minimum human involvement and can quickly test a large number of PCBs. Unlike the above mentioned PCB assurance methods which can be used either when a board is fully populated or unpopulated, AVI can potentially identify malicious inclusions, modifications and defects on PCBs at all stages of production and even after sale. It has many advantages over manual visual inspection which is slower, more expensive, less effective and subject to human error [171], [172].

Automated visual inspection can be implemented in three steps, image acquisition, image analysis and authentication (Figure 5.1) [5]. Image acquisition can be done using several imaging modalities [173], which can be categorised into three groups - surface, subsurface and volumetric. Depending on the requirements, multi-modal imaging approach may also be applied to detect, for example, malicious modifications between PCB layers or active components disguised as passive.

Collected images then need to be analysed for possible defects or malicious inclusions and modifications. Image analysis involves processing, feature extraction and classification stages. First the acquired images are processed to improve the quality, for example, by removing noise, altering illumination and enhancing contrast. The next stage is feature

extraction when key characteristics such as shape, color and texture of the pursued objects are captured. This is followed by classification and grouping of similar style components such as metal traces, vias, integrated circuits (IC), capacitors, transistors or resistors [122]. Classification traditionally was conducted by using template and feature matching methods to compare patterns and features between two images. With advances in deep learning methods, feature extraction and classification can be performed simultaneously using deep learning algorithms [5]. Text recognition is also used in classification for identification of markings such as serial numbers, which can be used, for example, for detection of counterfeit components by comparing the component's serial number with the manufacturer's original equipment serial number.

The final stage of automated visual inspection (AVI) is authentication where data is stored as a Computer Aided Design (CAD) file for comparing images of a fabricated PCB with the image of a golden PCB model [122].

Although AVI has been the most commonly used method for PCB assurance since 1960s [174] it had several limitations such as limited-area inspection, hard-coded specifications, and significant amount of subject matter expert involvement [148], [149]. Advances in deep learning in recent years, in particular image recognition, localization and segmentation, have been successfully applied to AVI to overcome these limitations. Several AVI methods have been suggested, including canonical image processing method for detection, classification and localisation of several specific types of defects on a PCB [123], convolutional neural network for detecting six types of defects [124], automated detection for component placement by directly comparing golden and test PCBs [125], [126] and text detection on the PCB for verification purposes [127], [128].

All the suggested approaches designed for PCB defect detection do not distinguish between irregularities that are due to manufacturing defects and malicious hardware Trojan (HT) inclusions, which are adversarial modifications for compromising sensitive information or causing denial of service. In the Big Hack [2], for example, an alien component implanted on a PCB was an HT which was visually disguised as one of the legitimate components, albeit being marginally larger in size. The HT was designed to provide administrative access to the network for an outside attacker. Development of AVI approaches for monitoring the location, size, and appearance of PCB components, focusing

in particular on HT detection, is very important [122]. This chapter addresses that problem by proposing a novel optical AVI algorithm for HT detection on a PCB.

5.2 Previous Work

In recent years AVI has been applied using several methods including image matching, feature extraction, and deep learning. Each of these approaches demonstrated effective performances in various defect detection tasks such as component, trace and via defect detection, and component classification.

Image matching methods have been mostly applied for detecting missing, displaced or replaced components [126], [175], [176], [177]. In particular, by using background subtraction 90% accuracy can be achieved on detecting the absence of capacitors and resistors [176]. By matching wavelet-transformed images, component inspection in electronic assembly lines is suggested by Cho et al. [177] with 86% accuracy. Feature extraction has also been adopted to classify component defects [125]. While image matching checks the whole PCB, feature extraction is applied only on regions where illegitimate components are expected to be present.

Feature extraction and deep learning methods have proven to be efficient for component classification [178], [179]. Youn et al. showed that an automatic surface mount device classification method can extract color and edge information from color images of PCB parts [180]. A neural network was used to classify chip-type packages and 97.6% average classification accuracy was achieved after adding additional edge information. Using a convolution neural network, some authors proposed a component classification method [181], [182]. Based on component images obtained from a PCB, the method suggested by Lim et al. [181] separated components from their backgrounds and classified them, achieving 90.8% accuracy. By training IC images collected online, IC components were identified with 92.3% accuracy in another research [182].

Previous works have looked into AVI for quality assurance, however HTs pose a separate challenge. The novelty in this work is that, instead of production fault detection, the algorithm has been optimised and trained specifically for implanted HT component

detection on a PCB. The proposed HT detection methodology has been trained and tested with three groups of HTs, categorised based on their surface area. The results show that it is possible to reach effective detection accuracy of 95.1% for HTs as small as 4 mm^2. In case of HTs with surface area larger than 280 mm^2 the detection accuracy is around 96.1%, while the average performance across all HT groups is 95.6%. These results can be further improved if higher resolution images are used.

The rest of the chapter is organised as follows: the proposed methodology is presented in section 5.3. Section 5.4 describes the experimental setup used for carrying out the research. Section 5.5 breaks down the proposed pipeline into three main stages and provides a motivational case study example. Discussion on results is carried out in section 5.6. Finally, the chapter is summarised in section 5.7.

5.3 Proposed Methodology

The goal of this work is to develop a low-cost and fast PCB visual inspection tool. This is achieved by avoiding expensive and slow imaging modalities such as X-rays or high end microscopes. Instead, the approach adopted in this chapter is detecting HT contaminated PCBs by using a simple digital optical camera. This fundamental characteristic of the proposed method allows to develop an AVI tool with marginal time and resource overheads, inspecting all PCBs passing through a conveyor belt setup on production lines.

The proposed methodology pipeline includes conventional computer vision techniques such as image alignment, blurring filters and background image subtraction and is comprised of the following key stages:

- Image alignment through homography matrix
- Application of Gaussian blurring filter
- Background image subtraction
- Suspicious region identification
- Cropping suspicious regions as image pairs
- Siamese Neural Network similarity estimation
- Confirmed dissimilar region marking on PCB

- Labelling the PCB on HT presence status.

5.3.1 Image Alignment through Homography Matrix

To align two images first a matrix, called the homography matrix, should be estimated. Then all pixel coordinates of one of the images should be multiplied by the homography matrix:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \sim \mathbf{H} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{5.1}$$

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}}, \quad y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}}, \tag{5.2}$$

where $\mathbf{H}$ is the estimated homography matrix, and (x', y') are the updated coordinate estimates of the pixel previously in location (x, y). It should be noted that this technique only works for images of two-dimensional flat surfaces. In the scope of this work, the aforementioned two-dimensional flat surface would be the PCB. Although a populated PCB has surface mount objects, e.g. capacitors, which protrude from the surface, in the scale of the whole PCB these are small deviations and can be disregarded. On the other hand, in the case of larger protrusions, such as a large heat-sink, the accuracy of the estimated homography matrix may suffer, hence this factor should be kept in mind and, where possible, the methodology should be applied prior to installation of such large components.

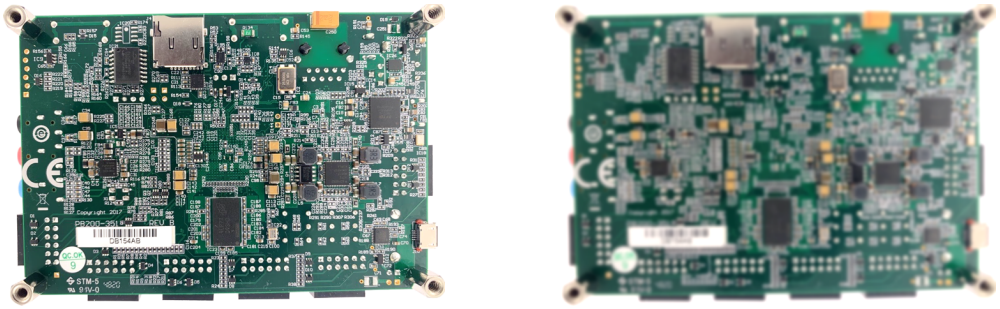
5.3.2 Application of Gaussian Blurring Filter

Following image alignment phase a blurring filter kernel is applied to both images. The visual effect of a blurring filter application is shown in Figure 5.2. This is done to smooth out minor misalignments on the edges of objects (e.g. wires, chips). The main idea behind this step is that if if there is an extra HT component of the PCB, then applying a soft blurring filter will not have an effect strong enough to lose the information about the presence of the HT. On the other hand, it could be the case that the differences between

the two images are only due to a slight misalignment, because of the marginal errors in the estimation of homography matrix. In this case, applying a blurring filter will remove these differences, leaving only the major differences, if they exist. In this work the kernel grid of the burring filter has been populated with a two-dimensional Gaussian distribution function $G(x, y)$, a.k.a. Gaussian blur filter:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \mathrm{e}^{-\frac{(x^2+y^2)}{2\sigma^2}}, \tag{5.3}$$

where $\sigma_x = \sigma_y = \sigma$ is the standard deviation on both axes.



(a) Original photograph. (b) Adjusted photograph

Figure 5.2: Photographs of a sample printed circuit board, before and after application of a blurring filter.

5.3.3 Background Image Subtraction, Suspicious Region Identification and Cropping as Image Pairs

Finally, image background subtraction and thresholding operations are demonstrated in Figure 5.3. In the process of background subtraction, the blurred image of the golden model (GM) PCB is subtracted from the blurred image of the PCB Under Inspection (PUI). The subtraction is a pixel-wise action, performed on pixels with same coordinates in the images. The result is a grayscale image referred to as the difference mask. The regions where pixel values of the two images did not match will be highlighted. The bigger the difference, the brighter white they will show on the difference mask, while similar looking regions will be marked in black. Since there is an inherent margin of error due to reasons such as change in lighting conditions (e.g. angle of incidence, brightness, color tone) when capturing images or the noise in camera sensors, to some degree even similar

looking regions on the PCB will be highlighted in white on the difference mask. That is why a threshold is applied on the pixel values in the difference mask to filter out the unwanted effects of noise. The final result is the thresholded difference mask with well defined contours. The resulting individual contours are regarded as suspicious regions and cropped out as separate images for later processing.

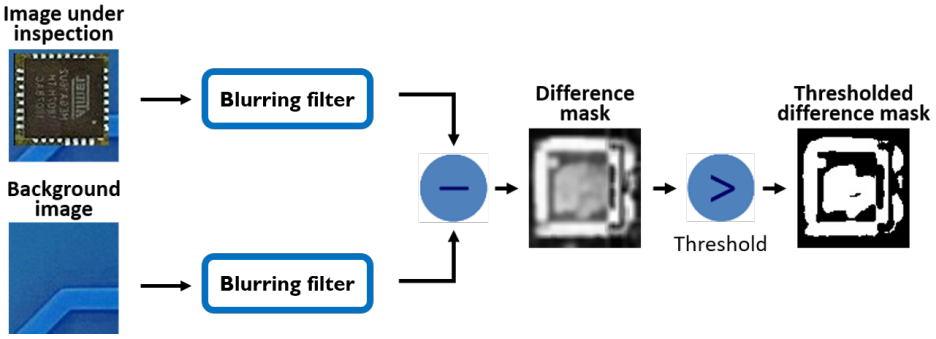


Figure 5.3: Image subtraction and pixel value binary thresholding.

5.3.4 Siamese Neural Network Similarity Estimation

The proposed methodology also includes deep learning architectures such as convolutional neural network and feed-forward fully connected neural network. These are merged into a specific architecture called Siamese Neural Network (SNN) [183], [184]. The SNN has a particular type of neural network architecture (Figure 5.4) where some weights are shared between two towers of convolutional neural networks. Each tower produces an embedding vector of its respective input image. Given a dataset of pairs of inputs, the network is trained to maximise the distance between the embeddings of the inputs coming from different classes, while minimizing the distance between embeddings coming from inputs of the same class. This process is referred to as supervised similarity learning.

In the scope of this work the input images to the twin towers of the SNN are the pairs of suspicious region images cropped out from the GM PCB and the PUI. The images are first separately processed by the convolutional neural network. Although in Figure 5.4 the diagram shows two convolutional neural networks, these have the exact same parameters, so essentially they can be treated a single network, receiving two inputs, one at a time, and generating an embedding for each one of them. Later these embeddings are subtracted and processed to acquire the similarity score.

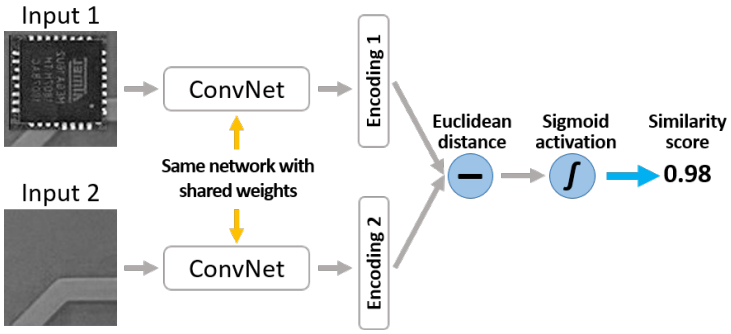


Figure 5.4: Siamese Neural Network architecture overview.

5.3.5 Controlled Environment for Capturing Photographs

In order to obtain a good quality dataset and minimise the ambient optical impact, the PCBs have been placed in a photographic tent with a built-in diffused light source (Figure 5.5a). This way, when capturing the photographs, all PCBs have similar initial environmental conditions, independent from many external factors such as daylight, shifting shadows, color variations due to reflections from the surroundings [185]. On the one hand it could be argued that outside the laboratory's controlled environment the ambient lighting conditions could vary. On the other hand, it can also be assumed that implementing a photographic tent-like structure in a fabrication facility production line can be achieved with little extra effort. The motivation for using a photographic tent to begin with is to boost the algorithm performance by removing unnecessary complications (e.g. changing light source color or angle of incidence). This is a low-effort, but high-impact improvement to the input data. Further, while producing the images the camera has been mounted on a static stand and a remote controlled shutter has been used to produce stable images.

5.4 Experimental Setup

5.4.1 Capturing Photographs

To build the PCB image dataset a $52cm \times 52cm \times 52cm$ FOSITAN photographic tent with an opening on the top and a built-in intensity-adjustable white LED light has been used. The intensity of the light source at the surface level of the PCB has been kept at

(a) Photographic tent used for creating the dataset of PCB images.

(b) Digital light meter used for controlling the light intensity.

Figure 5.5: Equipment used in creating the dataset of printed circuit board images.

5380 Lux, with the measurements being taken by an AP-8801C digital light meter (Figure 5.5b). Further, to minimise artifacts in the image, for example due to reflective surfaces on the PCB, several layers of light diffusing cloth sheets have been applied between the light source and the PCB, which is a standard practice for controlling image quality. Using this environment 101 images of a PCB have been captured of which 1 has been used as the golden model PCB and the other 100 were later used as source images for creating a much larger dataset of HT infected PCBs. Regarding the digital camera used to produce the images, a 12 Mp camera with a 1 cm sensor size and 1.4 μm pixel pitch has been utilised. The camera has a 26 mm equivalent focal length and $f/1.8$ aperture lens. The objective of this research is to develop a high quality automated visual inspection (AVI) algorithm which can work with moderate quality input images acquired through low cost digital optical camera modules.

5.4.2 Preprocessing Photographs

Before two images can be compared with each other to detect differences, they have to be aligned. This is a crucial step in the proposed AVI pipeline. During image alignment process one image is warped to match the second. This effect is achieved by multiplying the coordinates of every pixel in the image by the homography matrix **H** as shown in Eq. (5.1) on page 103. It is important to note that homography can only be applied to objects on a 2D plane such as a PCB. In this research, OpenCV library has been used to compute the homography matrix and perform image alignment [186].

In order to perform homography, first the same points of the object (e.g. PCB) in both images need to be located. The minimum required number of such point pairs to be able to perform homography is four. Increasing the number of such pairs will result in a more robust homography estimation and, hence, improved alignment of images. To acquire such point pairs in an automated manner, first several key-points on both images need to be located. Then these key-points should be matched into pairs. For example, it can be achieved with brute force matching by searching for the minimum euclidean distance between every couple of descriptor vectors belonging to a particular pair of key-points. Points referred to as key-points are typically distinctive corners, edges or sharp curves of the objects (e.g. PCB) present in the image. They are defined by (x, y) coordinates, size and orientation. Their respective descriptors, on the other hand, are unique markers of the key-points, independent from the orientation of the object in the image. The descriptors are vectors calculated internally by OpenCV [186] and help in search for matching key-point pairs from two different images of the same item.

At a later stage, another algorithm called Random Sample Consensus (RANSAC) is used to discard the key-point pairs with a high likelihood of being outliers. Such points can have a significant negative impact on the quality of the homography matrix. RANSAC is an iterative algorithm which validates a mathematical model built using a dataset with outliers [187]. The algorithm assumes that the dataset is a combination of both inliers and outliers. Inliers can be identified by a model with a special set of parameter values, whereas outliers do not fit the model in any condition. The iteration process repeats a fixed number of times and each time produces either a model which is rejected due to very few points being part of the consensus set, or produces a refined model together with a corresponding updated consensus set. The refined model is accepted only if the size of the updated consensus set is larger then that of the previous model.

5.4.3 Inserting Hardware Trojans

Three groups of HTs have been used in this research. They have been binned into groups of small, medium and large, based on their surface area (Table 5.1).

The HTs have been added into the dataset with the help of Flip library on GitHub developed by LinkedAI [188]. The library is used for synthetic data generation on new

Table 5.1: Groups of hardware Trojans.

Group name	Small	Medium	Large
Surface area in mm^2	4 to 9	15 to 50	280+
Surface area in pixels^2	700 to 1500	2500 to 9000	50000+

2D images from a batch of objects and backgrounds. In the scope of this research, the background image is an HT free PCB Under Inspection (PUI), while the objects are the HTs. The idea is to take a random background image and a random object and place the object in a random location with a random integer multiple of 90° rotations. On top of that, the Flip library provides many of the conventional image augmentation functionalities, e.g. resize or colour shift the objects. Using Flip library and the 100 source images captured earlier as backgrounds, 7,500 images of PCBs with inserted HT devices have been generated, i.e. 2,500 images per HT group.

5.5 Hardware Trojan Detection Pipeline

The proposed HT detection pipeline consists of three main stages shown in Figure 5.6. In the first stage the images of the golden model (GM) and a PCB under inspection (PUI) are compared to identify the suspicious regions on the PUI, where an HT could be present, but in this stage there is no definitive prediction whether that is the case. Instead, the information on these regions is passed forward to the next stage as a list of bounding boxes.

In the second stage of the pipeline the algorithm uses the bounding boxes to crop out these sections as a set of smaller images. This step is repeated for both the GM and the PUI to create pairs of images of every suspicious region.

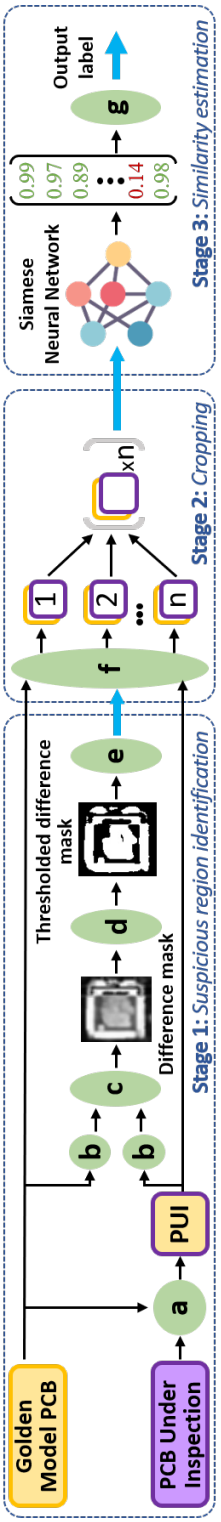


Figure 5.6: The proposed hardware Trojan detection pipeline, where: (a) Image alignment with homography, (b) Gaussian blurring kernel, (c) Background image subtraction, (d) Binary thresholding, (e) Suspicious region identification, (f) Cropping suspicious regions as image pairs from GM and PUI, (g) Labelling the PUI on HT presence status.

A more detailed illustration of the second stage of the pipeline is provided in Figure 5.7. As it can be seen in Figure 5.7, there are three steps involved in the second stage of the pipeline. First all suspicious regions are cropped out as new individual images. This step is repeated for the GM and PUI to create a pair of images per suspicious region. Next, since these new smaller image pairs will be of varying shapes and sizes dictated by the bounding boxes determined in the previous stage of the pipeline, they are all normalised to a predetermined shape, which in this case is 28×28 pixels. Finally, the already normalised images of the suspicious regions are converted to grayscale, in preparation for the next stage of the pipeline.

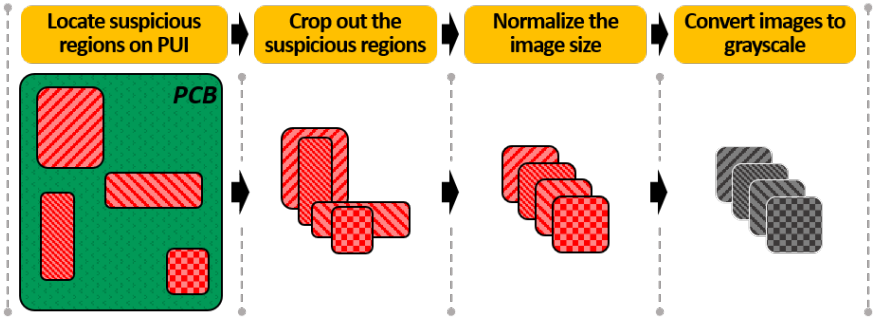


Figure 5.7: Cropping and normalisation of suspicious regions.

The resulting image pairs of the suspicious regions are passed on to the final third stage of the pipeline in Figure 5.6 - the Siamese Neural Network (SNN). Normalisation of the input image sizes is necessary for the SNN to function, since it can only work with a specific input shape, predefined by the network architecture at the time of model training. Next, converting the input images to grayscale is done to reduce the size of the neural network, by reducing the number of required trainable parameters. After receiving all normalised image pairs of the suspicious regions, the SNN individually processes every pair, outputting a similarity score for each pair.

In this work, the Keras framework [189] has been used for implementing the SNN. The network is trained to differentiate between the HT contaminated and HT clean image pairs. In case of the HT clean image pair class the two cropped images from GM and PUI should be very similar. The reason why such regions have been suggested by the previous stage of the pipeline is that although the image alignment algorithms have a good performance, they are not perfect and in some cases a slight misalignment gets interpreted

as a physical difference on the PCB. This is where the SNN excels at differentiating between a misalignment and actual HT presence. Finally a threshold is applied to the similarity scores followed by a logic AND function to check if at least one HT is present on the PUI.

5.5.1 Case Study

To explain the proposed methodology, a simplified case study can be considered, where 1000 PUIs are available, 500 out of which have an HT on board (Figure 5.8). The first stage of the proposed pipeline in Figure 5.6, the suspicious region identification stage, is optimised to mark in suspicious regions as many of the 500 HTs as possible, even though in the process the algorithm may also wrongly suggest a large number of misidentified suspicious regions. For example, the algorithm may miss 10 HTs on HT infected PUIs, correctly mark 490 HT containing regions and further suggest 260 misidentified regions on the PUIs, which do not contain an HT. These 750 suspicious region coordinates are cropped from the their respective PUIs, as well as the golden model (GM). They are later normalised as described in Figure 5.7 and passed to the SNN in the last stage of the pipeline in Figure 5.6. The SNN individually compares all 750 image pairs to assess their similarity. For example, the SNN may have 95% accuracy, detecting 465 out of 490 HTs. In such a scenario, 490 out of 500 HTs were detected by the first stage (98% detectability) and 465 out of those 490 HTs were detected in the final stage (95% accuracy) resulting in an overall 93% effective accuracy.

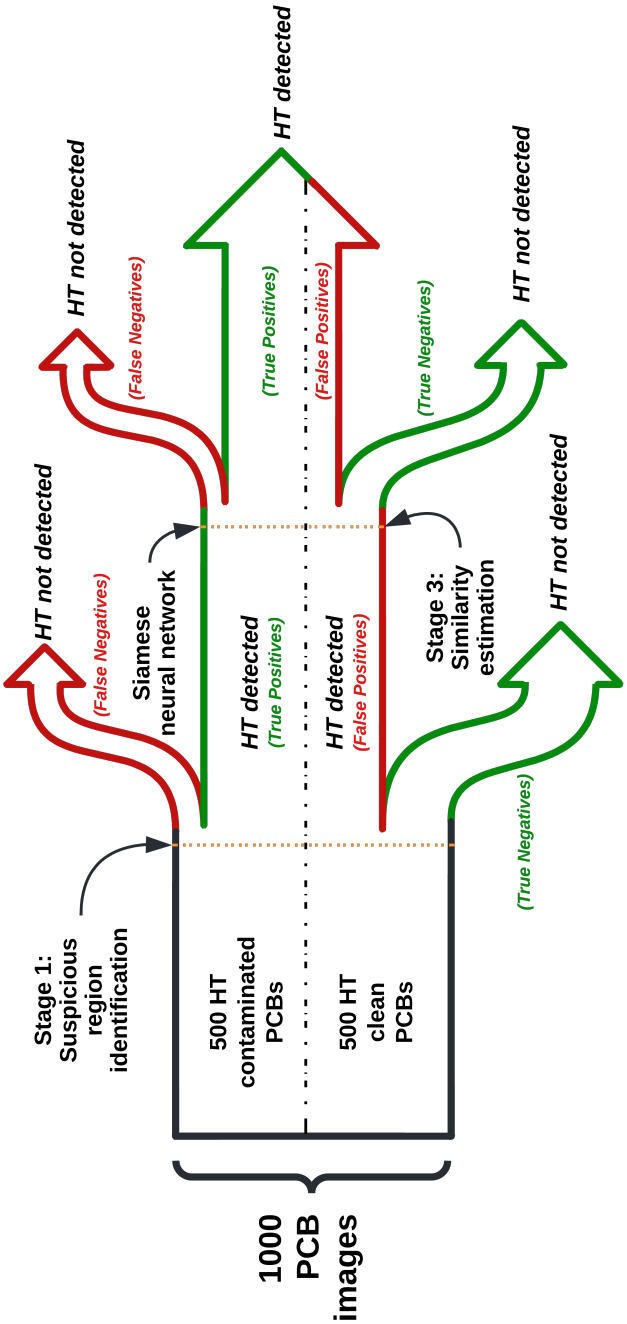


Figure 5.8: Motivational example illustrating the prediction flow for 500 HT contaminated and 500 HT clean PCB images. The diagram does not represent the real HT detection results in proportion and is rather intended as a guidance in understanding the process.

5.6 Experimental Results

Since the proposed methodology is organised in a multistage setup (Figure 5.9), it is possible to retrieve meaningful outputs from the intermediate stages. In fact, this is a crucial step in the overall optimisation process. In this work the methodology has been optimised for two independent consecutive results, i.e. *image thresholding* and *siamese neural network*.

5.6.1 Image Thresholding

The stage of suspicious region identification through image thresholding has undergone a constrained optimisation problem, whereby the HT detection rate, i.e. detectability, has been maximised, such that the ratio of misidentified suspicious regions to the total number of predicted suspicious regions does not exceed 95%. Here detectability is defined by the ratio of all HTs which were included in at least one of the suggested suspicious regions. The suspicious region is defined as misidentified if it does not overlap with an HT or if the intersection over union (IOU) is below 10%. In other words, this algorithm has been optimised to find as many HT containing regions as possible, while keeping the rate of wrongly suggested suspicious regions in a reasonable range. This constraint on the optimisation has been introduced to avoid the trivial case of having the algorithm mark all of the PCB surface as suspicious. The optimisation was accomplished by calibrating the pixel value cutoff threshold for image binary thresholding (Figure 5.10) using the 1 GM and 7,500 PUI images with their ground truth masks of HT locations. The results for HT detectability, alongside with the respective rate of misidentified proposed suspicious regions, are presented in Table 5.2, subject to varying cutoff thresholds. The cells satisfying the constraint of keeping the rate of misidentified suspicious regions below 95% have been highlighted with a light green background. The reason why the algorithm can afford to output so many misidentified suspicious regions is that the SNN in the last stage of the pipeline (Figure 5.9) can discard them with high accuracy. The goal here is to mark as many HT containing regions as possible.

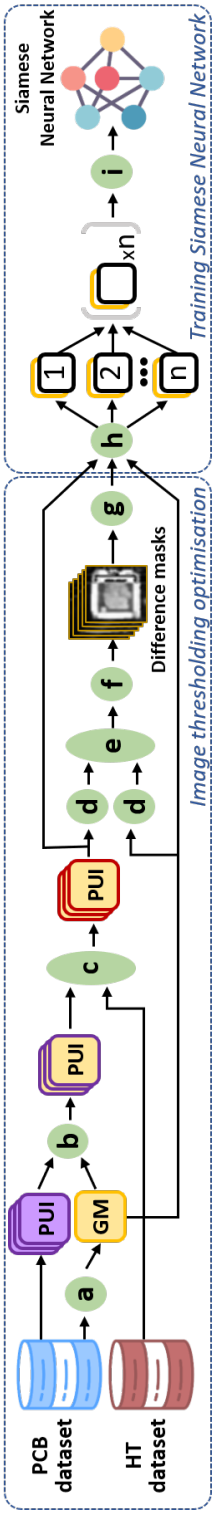


Figure 5.9: Training process of the proposed methodology, where: (a) Choose one as the golden model (GM), (b) Align the remaining images to the GM, (c) Insert HTs on every PCB under inspection (PUI), (d) Gaussian blurring kernel, (e) Subtract GM from every PUI, (f) Binary thresholding, (g) Suspicious region identification, (h) Cropping suspicious regions as image pairs from their respective pre-blurred PUI and GM images, (i) Train Siamese Neural Network.

Table 5.2: Hardware Trojan detectability in top-left blue and rate of misidentified suspicious regions in bottom-right red.

Cutoff threshold	Small HT	Medium HT	Large HT
15	99.6% / 98.1%	99.1% / 95.9%	100% / 98.0%
25	99.4% / 96.2%	99.4% / 91.2%	100% / 95.7%
35	99.6% / 94.4%	99.7% / 87.6%	100% / 93.9%
45	99.6% / 93.0%	99.6% / 84.8%	100% / 92.3%
55	93.4% / 88.4%	98.2% / 76.8%	100% / 87.3%

(a) *Source img.* (b) *thr.* = 150 (c) *thr.* = 100

Figure 5.10: Given a grayscale source image (a), computed binary thresholding images with high (b) and low (c) threshold values.

5.6.2 Siamese Neural Network

A Siamese Neural Network with around 872×10^3 trainable parameters has been trained to discern between the images of the same patches on two PCBs harbouring an HT component, while being able to recognise similar patches which are only slightly misaligned or misshaped (Figure 5.11). The root cause of having such misaligned patches is the estimation of the homography matrix **H** in Eq. (5.1) on page 103. The reason for having misshaped patterns could be, among other things, variations in the PCB production process as well as defects such as misaligned elements.

Information about the datasets used to train, validate and test the SNN model and their respective resulting prediction accuracies are shown in Table 5.3. The datasets are comprised of pairs of images, where each pair represents one of the suspicious regions. Inside every pair the first image is cropped from the suspicious region on the PCBs under inspection, while the second is the matching region on the golden model PCB. These images are of the exact same regions on both PCBs, cropped out based on coordinates

(a) *Golden model PCB* (b) *PCB under inspection* (c) *Golden model PCB* (d) *PCB under inspection*

Figure 5.11: Cropped suspicious regions. Image pair (a) and (b) are only misaligned, pair (c) and (d) harbour a Trojan component.

Table 5.3: Siamese Neural Network prediction accuracy.

Dataset name	**Train**	**Validation**	**Test**
Dataset size	18000	6000	6000
Prediction accuracy (*small HT*)	98.8%	96.5%	**95.5%**
Prediction accuracy (*medium HT*)	98.4%	97.6%	**95.9%**
Prediction accuracy (*large HT*)	98.7%	98.5%	**96.1%**

from a single bounding box and later normalised as shown in Figure 5.7. In total about $30K$ such image pairs have been collected and split into training, validation and testing datasets with a $60\% - 20\% - 20\%$ ratio. The performance of SNN has been analysed per HT group (Table 5.3). The classification accuracies on testing dataset range from 95.5% to 96.1%, as expected, performing best on large HTs.

5.6.3 Effective Accuracy

Since the proposed methodology has two consecutive, distinct and independent outputs with their respective accuracy scores, the effective accuracy of the methodology as a whole is the multiplication of the two. In other words, HT detection accuracy of the SNN applies only to the HTs which had previously been detected by the previous stage of the pipeline. For example, in case of medium size HTs the suspicious region identification with image thresholding stage resulted in HT detectability rate of 99.7% and the SNN had classification accuracy of 95.9%. The resulting effective accuracy of the methodology for medium size HTs is $(99.7\% \times 95.9\%) = 95.6\%$. The effective accuracies for all groups of

Table 5.4: Effective prediction accuracy.

HT size group	Small	Medium	Large	**All**
Image thresholding *(at 35)*	99.6%	99.7%	100%	**99.8%**
Siamese Neural Network	95.5%	95.9%	96.1%	**95.8%**
Effective accuracy	**95.1%**	**95.6%**	**96.1%**	**95.6%**

HTs are presented in Table 5.4. As expected, the algorithm performance improves up to 96.1% as the HTs get larger, with the overall HT implanted PCB detection accuracy being around 95.6%.

5.7 Concluding Remarks

This chapter proposes a methodology for detecting hardware Trojan (HT) components on a printed circuit board (PCB) through automated visual inspection. It is assumed that an image of a trusted golden model (GM) of the PCB is available with which comparisons are made. The proposed technique provides an accurate and fast tool to detect HT inclusions on PCBs using a low-cost imaging modality - optical digital camera. To keep the operating conditions stable and avoid the negative impacts from variations in ambient lighting, a photographic tent (Figure 5.5a) with internal diffused light source has been used to develop a PCB image dataset containing $7,500$ plus 1 images including the GM.

The proposed methodology is a pipeline of three sub-stages (Figure 5.6). Initially, the first stage proposes suspicious regions on the PCB Under Inspection (PUI), where a potential HT could be located. In the second stage, these regions are cropped out as pairs of images from both the GM and PUI. In total 30,000 such pairs of images are preprocessed (Figure 5.7) preparing them for the final stage. In the final stage, a Siamese Neural Network (SNN) takes each of these 30,000 image pairs as two separate inputs and outputs a similarity estimation. The results show that the proposed automated visual inspection pipeline, combining conventional computer vision techniques and deep learning, can detect HT devices with surface area from 4 mm^2 to 280 mm^2 implanted on a PCB with an effective accuracy of 95.6% (Table 5.4). Thus, the objectives set out in section 2.7 concerning the third contribution have been met in this chapter.

Chapter 6

Conclusions

The continuous integration of increasingly more electronic devices into various aspects of modern society has undoubtedly been beneficial in the most part. Technology has led to historically highest labour productivity and quality of life. However, these trends of widespread technology adoption have also led to ever higher demands on the market, putting original equipment manufacturers and design houses under pressure to cut time to market and devise alternative strategies to meet the demand. Outsourcing the production to other companies is now a common industry practice. Unfortunately, outsourcing to untrusted fabrication facilities creates hardware security risks, threatening with unanticipated negative consequences, particularly in safety critical applications. One of the most prominent types of hardware security threats are hardware Trojans (HTs). The research in this thesis has investigated methodologies directed at detection of HT implants on a Printed Circuit Board (PCB). The objective of the thesis was to address the lack of PCB level HT detection methods by proposing and experimentally validating two power analysis based and one computer vision based techniques, which are summarised in the next section followed by the proposed future work.

6.1 Thesis Contributions

The overall goal of this thesis has been to develop methods for board level hardware Trojan detection. This goal is achieved through three main projects. The first two projects are based on anomaly detection through power analysis of the PCB components, while the third project uses computer vision and deep learning to detect extra HT components added on the surface of the PCB. The proposed methodologies are directed towards detecting added HT components on the PCB, such as a rogue microcontroller, and are independent

of the specific trigger and payload functionality of the Trojan.

The objective of the first project was to detect HTs on the PCB, which draw power from the power distribution network. In Chapter 3, a novel methodology is proposed on detecting HT components on a PCB using power monitoring. The experimental results show that the proposed Differential Power Monitoring (DPM) method can detect Trojans implanted on the PCB with low false positive rate provided an appropriate detection threshold. The differential power monitoring technique provides additional protection for end users without affecting the throughput of the PCB.

In the second project the monitoring circuit architecture proposed in Chapter 3 is reused, adding extra features with the introduction of ML methods. The objective in Chapter 4 was to detect stealthier HTs powered from the input/output ports of legitimate chips on a PCB. Such HTs can circumvent the HT detection methodology proposed in Chapter 3, hence the requirement for this second project.

Two ML algorithms have been applied on the power consumption data collected from a purpose built PCB prototype. This dataset has been injected with one hundred categories of HTs. One-Class Support Vector Machine (SVM) and Local Outlier Factor (LOF) one class classification algorithms have been used to train machine learning models capable of detecting anomalies in the power consumption patterns across the PCB. Simulations for both algorithms returned classification results reaching F1-scores above 99.5% and 97% for One-Class SVM and LOF respectively, given the HT's mean power consumption is $\bar{P}_{HT} \geq 45\ mW$.

The simulation results have been validated through real-life experiments carried out on the prototype PCB. Comparison between simulation and experimental results has been presented for nine hardware Trojan categories, their power consumption ranging from 5 mW to 20 mW. Further, the One-Class SVM model is low-cost and requires only 27 KB memory to operate.

The objective of the third project in Chapter 5, was to propose a methodology for detecting HT components on a PCB through automated visual inspection (AVI). It is assumed that an image of a trusted golden model (GM) of the PCB is available with which comparisons are made. The proposed technique provides an accurate and fast tool to detect HT inclusions on PCBs using a low-cost imaging modality - optical digital camera.

The proposed methodology is a pipeline of three sub-stages. The first stage proposes suspicious regions on the PCB Under Inspection (PUI), wherever there is an increased possibility of a potential HT being located. In the second stage, these regions are cropped out as pairs of images from both the GM and PUI. To train the deep learning model a total of 30,000 such pairs of images are preprocessed, preparing them for the final stage. In the final stage, a Siamese Neural Network (SNN) is trained on each of these 30,000 image pairs as two separate inputs to output a similarity estimation for every pair. The results show that the proposed automated visual inspection pipeline, combining conventional computer vision techniques and deep learning, can detect HT devices with surface area from 4 mm^2 to 280 mm^2 implanted on a PCB with an effective accuracy of 95.6%.

In conclusion, this thesis provides a novel contribution to the subject of PCB level HT detection. The research outcomes of this thesis have met the main objectives laid out in the introduction, that is to contribute in developing board level HT detection methodologies.

6.2 Limitations

Detectability of HTs can be affected by issues that were overlooked or were beyond the scope of this work. The following bullet points reflect on such research gaps.

- The detectability of HTs is also affected by issues such as the RLC characteristics of wires on the PCB or PCB workload-dependent drift in parasitic power consumption of the power distribution network (PDN). For example, the capacitive and inductive components on the PDN can frequently induce sharp rises to parameter P_{PDN} in Chapter 3, section 3.4, ultimately having an adverse effect on the false positive rate and accuracy of the methodology. Even in the unrealistic noise-free environment, using state-of-the-art power sensors, these effects will provide a window of opportunity for the adversary to implant an HT with a small side-channel footprint and evade detection. Therefore, RLC effects on the proposed methodologies should be analysed, alongside with a deeper investigation into their theoretical underpinning and novel techniques should be developed to solve the problem. One such possible solution to this problem is proposed in section 6.3.

- Using a faster communication protocol to communicate data between the power sensors and the main monitoring computational block on the PCB should improve the results presented in work. It would allow the methodologies presented in Chapter 3 and Chapter 4 to operate at a higher frequency, pushing down the range of minimum detectable HT activation time. For example, in Chapter 3 the operating frequency of the monitoring setup on the prototype PCB was around $\nu = 100\ Hz$. This set the threshold of minimum detectable HT activation period at about 10 ms. The power sensors used in the prototype were limited to a relatively slow Inter-Integrated Circuit (I^2C) (up to 2.56 Mbps *High-speed* mode) communication protocol. Furthermore, the resolution of power sensors used in this work is limited to 1 mW. It is clear that even lower power HTs can be detected by changing the monitoring equipment and introducing more accurate sensors with better resolution. For example, using INA226 digital power sensors with a resolution of 10 μW (ADC 16 bits) will decrease the lower limit of detection threshold up to 100 times, compared to the setup in current PCB prototype.
- In Chapter 4, it was assumed that an inclusive power consumption dataset from the golden model PCB, reflecting all of its legitimate operational modes, is available to use for training the ML model. However, it could often prove to be the case, that it is not possible to build such an inclusive dataset due to the multitude of the legitimate operation combinations on the PCB. In this case, novel approaches would be required to either synthetically adapt the existing data with further assumptions or find a new way to train the machine learning model. Either way, further investigation is required to address this scenario.

6.3 Future Work

Despite all the existing scientific contributions, be it within this thesis or otherwise, research on hardware Trojan detection is still in its infancy and significant novel academic research is still necessary. To address this, some possible future research directions are provided in the following bullet points. In general, the target of future research should be

a more robust and reliable family of methodologies. These should accommodate various techniques under a single umbrella approach, simultaneously combating many different subgroups of HTs. These future improved methodologies should also resolve problems related to issues such as process, voltage or temperature variation. In this manner, being closer to fundamentally solving hardware security problems related to Trojan implants, the research will become more attractive for industrial exploitation, crossing the chasm between being a pure research topic and a real-life benefit. In addition, future research should also be directed at developing the necessary research infrastructure, such as related software tools and open source datasets aimed at assisting the wider research community, facilitating further developments in the field. Possible directions of future research include the following:

- The capacitive and inductive characteristics of the power distribution network (PDN) can have a significant impact on its parasitic power consumption in the form of sharp rises. The methodology proposed in Chapter 3 should be equipped to differentiate such *acceptable* peaks on the ΔP_{HTinf} (Eq. (3.4)) time series from actual HT peaks. In that regard, a 1-dimensional convolutional neural network (CNN) can be trained on the ΔP_{HTinf} time series data from the golden model PCB, to learn the shapes of these RLC induced peaks and identify anomalous peaks introduced by the HT's power consumption. Such 1-D CNN can become part of the existing proposed methodology and tackle the problem of RLC effects.
- Improving the sensors used for harvesting information about the state of the PCB. Using more sophisticated sensors should improve the results reached in Chapter 3 and Chapter 4. Sensor parameters such as resolution, operating frequency and accuracy set a limitation on minimum detectable HT specifications. This relatively simple improvement should push the boundaries of HT detectability suggested in this work. For example, using INA226 digital power sensors with a resolution of 10 μW (ADC 16 bits) will decrease the threshold of minimum detectable HT power consumption around 100 times compared to the setup used on the prototype discussed in this work. Improving resolution of the sensors will also help reduce the noise-to-signal ratio, improving data quality and thus the results in Chapters 3 and 4.
- Variabilities such as voltage, process and temperature variation or aging of silicon

devices are another set of factors, which can have a negative impact on the performance metrics of the proposed methodology. Thus, incorporating these phenomena in future assessments is important. While voltage and temperature variations can become part of the current model in Chapter 4 as two more feature columns in the dataset, addressing process variation and aging may require a broader approach. For example, process variation can be addressed by collecting data from a wider range of PCB prototypes. Aging, on the other hand, can in theory be addressed by developing a number of models, which will all be uploaded on the device. Then, the primary active model, performing the monitoring, can be consecutively updated by one of these models based on the state of PCB aging [190], to reflect the new state of the PCB's power consumption.

- The surface area of the smallest HT used in the analyses in Chapter 5 was 4 mm^2. Developing the results reached in Chapter 5, detection of ultra-small HT components, with surface areas smaller than the ones used in this research, should be analysed in a future work.

References

[1] K.Rebello, "Supply chain vulnerabilities," in *FICS Research Annual Conference on Cybersecurity*, 2019.

[2] J., Robertson and M., Riley, "The big hack: How china used a tiny chip to infiltrate u.s. companies," https://bloomberg.com/news/features/2018-10-04/the-big-hack-how-china-used-a-tiny-chip-to-infiltrate-america-s-top-companies/, 2018.

[3] https://www.supermicro.com/en/news/CEO-letter/.

[4] https://media.ccc.de/v/35c3-9597-modchips_of_the_state/.

[5] D. Mehta, H. Lu, O. P. Paradis, M. M. Azhagan, M. T. Rahman, Y. Iskander, P. Chawla, D. L. Woodard, M. Tehranipoor, and N. Asadizanjani, "The big hack explained: Detection and prevention of PCB supply chain implants," *J. Emerg. Technol. Comput. Syst.*, vol. 16, no. 4, 2020. [Online]. Available: https://doi.org/10.1145/3401980

[6] K. Rosenfeld and R. Karri, "Attacks and defenses for jtag," *IEEE Design Test of Computers*, vol. 27, no. 1, pp. 36–47, 2010. [Online]. Available: https://doi.org/10.1109/MDT.2010.9

[7] A. Hennessy, Y. Zheng, and S. Bhunia, "JTAG-based robust PCB authentication for protection against counterfeiting attacks," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2016, pp. 56–61. [Online]. Available: https://doi.org/10.1109/ASPDAC.2016.7427989

[8] S. Paley, T. Hoque, and S. Bhunia, "Active protection against PCB physical tampering," *2016 17th International Symposium on Quality Electronic Design (ISQED)*, pp. 356–361, 2016. [Online]. Available: https://doi.org/10.1109/ISQED.2016.7479227

[9] W. Jillek and W. K. C. Yung, "Embedded components in printed circuit boards: A processing technology review," *International Journal of Advanced Manufacturing Technology*, vol. 25, no. 3–4, pp. 350–360, 2005. [Online]. Available: https://doi.org/10.1007/s00170-003-1872-y

[10] S. Ghosh, A. Basak, and S. Bhunia, "How secure are printed circuit boards against trojan attacks?" *IEEE Design Test*, vol. 32, no. 2, pp. 7–16, 2015. [Online]. Available: https://doi.org/10.1109/MDAT.2014.2347918

[11] N. Mar, P. Yarlagadda, and C. Fookes, "Design and development of automatic visual inspection system for PCB manufacturing," *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 5, pp. 949–962, 2011. [Online]. Available: https://doi.org/10.1016/j.rcim.2011.03.007

[12] M. McGuire, U. Ogras, and S. Ozev, "PCB hardware trojans: Attack modes and detection strategies," in *IEEE VLSI Test Symposium (VTS)*. IEEE, 2019, pp. 1–6. [Online]. Available: https://doi.org/10.1109/VTS.2019.8758643

[13] N. Sklavos, R. Chaves, G. D. Natale, and F. Regazzoni, *Hardware Security and Trust*. Springer, 2017.

[14] Y. Jin and Y. Makris, "Hardware trojan detection using path delay fingerprint," *IEEE International Workshop on Hardware-Oriented Security and Trust*, pp. 51–57, 2008. [Online]. Available: https://doi.org/10.1109/HST.2008.4559049

[15] M. Potkonjak, A. Nahapetian, M. Nelson, and T. Massey, "Hardware trojan horse detection using gate-level characterization," in *Proceedings of the 46th Annual Design Automation Conference*, 2009, p. 688–693. [Online]. Available: https://doi.org/10.1145/1629911.1630091

[16] M. Banga and M. S. Hsiao, "A novel sustained vector technique for the detection of hardware trojans," in *in Proc. of the International Conference on VLSI Design*, 2009, pp. 327–332. [Online]. Available: https://doi.org/10.1109/VLSI.Design.2009.22

[17] T. J. Oliveira, M. A. Wehrmeister, and B. T. Nassu, "Detecting modifications in printed circuit boards from fuel pump controllers," in *Conference on Graphics, Patterns and Images*. SIBGRAPI, 2017, pp. 87–94.

[18] "Nsa ant catalog, electronic frontier foundation," 2013. [Online]. Available: https://www.eff.org/document/20131230-appelbaum-nsa-ant-catalog

[19] J. Boone, "Tpm genie: Interposer attacks against the trusted platform module serial bus," Tech. Rep., 2018.

[20] J. Harrison, N. Asadizanjani, and M. Tehranipoor, "On malicious implants in PCBs throughout the supply chain," *Integration*, vol. 79, pp. 12–22, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167926021000304

[21] A. Iyengar and S. Ghosh, *Hardware Trojans and Piracy of PCBs*. Cham: Springer International Publishing, 2018, pp. 125–145. [Online]. Available: https://doi.org/10.1007/978-3-319-68511-3_6

[22] M. Tehranipoor and F. Koushanfar, "A survey of Hardware Trojan taxonomy and detection," *IEEE Design Test of Computers*, vol. 27, no. 1, pp. 10–25, 2010.

[23] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan, "Hardware trojan attacks: Threat analysis and countermeasures," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1229–1247, 2014. [Online]. Available: https://doi.org/10.1109/JPROC.2014.2334493

[24] M. Dhwani, L. Hangwei, P. Olivia, A. Mukhil, R. Tanjidur, I. Yousef, C. Praveen, W. Damon, T. Mark, and A. Navid, "The Big Hack explained: Detection and prevention of PCB supply chain implants," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 16, no. 4, pp. 1–25, 2020. [Online]. Available: https://doi.org/10.1145/3401980

[25] K. Xiao, D. Forte, Y. Jin, R. Karri, S. Bhunia, and M. Tehranipoor, "Hardware trojans: Lessons learned after one decade of research," *ACM Transactions on Design Automation of Electronic System*, vol. 22, no. 1, pp. 1–23, 2016. [Online]. Available: https://doi.org/10.1145/2906147

[26] T. Hoque, S. Yang, A. Bhattacharyay, J. Cruz, and S. Bhunia, "An automated framework for board-level trojan benchmarking," 2020. [Online]. Available: https://arxiv.org/abs/2003.12632

[27] K. Yang, M. Hicks, Q. Dong, T. Austin, and D. Sylvester, "A2: Analog malicious hardware," in *2016 IEEE Symposium on Security and Privacy (SP)*, 2016, pp. 18–37. [Online]. Available: https://10.1109/SP.2016.10

[28] X. Wang, M. Tehranipoor, and J. Plusquellic, "Detecting malicious inclusions in secure hardware: Challenges and solutions," in *IEEE International Workshop on Hardware-Oriented Security and Trust*. IEEE, 2008, pp. 15–19.

[29] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware trojans," *Computer*, vol. 43, no. 10, pp. 39–46, 2010. [Online]. Available: https://doi.org/10.1109/MC.2010.299

[30] H. Salmani, *Trusted Digital Circuits*. Springer, 2018.

[31] P. Miller, "A tale of four substrates-emi and thermal comparisons of different PCB materials," in *International Conference on Electromagnetic Compatibility, 1997. (Conf. Publ. No. 445)*, 1997, pp. 125–130. [Online]. Available: https:10.1049/cp:19971131

[32] A. Greenberg, "Planting tiny spy chips in hardware can cost as little as $200, wired," Tech. Rep., 2019. [Online]. Available: https://www.wired.com/story/plant-spy-chips-hardware-supermicro-cheap-proof-of-concept/

[33] S. Wakabayashi, S. Maruyama, T. Mori, S. Goto, M. Kinugawa, Y.-i. Hayashi, and M. Smith, "A feasibility study of radio-frequency retroreflector attack," in *USENIX Workshop on Offensive Technologies (WOOT 18)*, 2018.

[34] J. Winter and K. Dietrich, "A hijacker's guide to communication interfaces of the trusted platform module," *Computers Mathematics with Applications*, vol. 65, no. 5, pp. 748–761, 2013, eUROPKI-2011. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0898122112004634

[35] B. Blaxhil and J. Sandin, "Picodma: Dma attacks at your fingertips," 2019. [Online]. Available: https://i.blackhat.com/USA-19/Wednesday/us-19-Sandin-PicoDMA-DMAAttacks-At-Your-Fingertips.pdf.

[36] R. Gonggrijp and W. Hengeveld, "Studying the nedap/groenendaal es3b voting computer: A computer security perspective," in *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology*, ser. EVT'07, 2007, p. 1.

[37] S. Bhunia and M. Tehranipoor, *The Hardware Trojan War Attacks, Myths, and Defenses*. Springer, Cham, 2018. [Online]. Available: https://doi.org/10.1007/978-3-319-68511-3

[38] H. Salmani, M. Tehranipoor, and R. Karri, "On design vulnerability analysis and trust benchmarks development," in *International Conference on Computer Design (ICCD)*, 2013, pp. 471–474. [Online]. Available: https://doi.org/10.1109/ICCD.2013.6657085

[39] H. Salmani and M. Tehranipoor, "Trust-hub," 2013. [Online]. Available: https://trust-hub.org/home

[40] "Kicad eda, a cross platform and open source electronics design automation suite." https://www.kicad.org/.

[41] H. Zhu, X. Guo, Y. Jin, and X. Zhang, "PCBench: Benchmarking of board-level hardware attacks and trojans," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2021, pp. 396–401. [Online]. Available: https://doi.org/10.1145/3394885.3431596

[42] K. G. Liakos, G. K. Georgakilas, S. Moustakidis, N. Sklavos, and F. C. Plessas, "Conventional and machine learning approaches as countermeasures against hardware trojan attacks," *Microprocessors and Microsystems*, vol. 79, p. 103295, 2020. [Online]. Available: https://doi.org/10.1016/j.micpro.2020.103295

[43] D. Agrawal, S. Baktir, D. Karakoyunlu, P. Rohatgi, and B. Sunar, "Trojan detection using ic fingerprinting," *IEEE Symposium on Security and Privacy (SP '07)*, pp. 296–310, 2007. [Online]. Available: https://doi.org/10.1109/SP.2007.36

[44] J. Aarestad, D. Acharyya, R. Rad, and J. Plusquellic, "Detecting trojans through leakage current analysis using multiple supply pad $i_{DDQ}s$," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 893–904, 2010. [Online]. Available: https://doi.org/10.1109/TIFS.2010.2061228

[45] R. Rad, J. Plusquellic, and M. Tehranipoor, "A sensitivity analysis of power signal methods for detecting hardware trojans under real process and environmental conditions," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 18, no. 12, pp. 1735–1744, 2010. [Online]. Available: https://doi.org/10.1109/TVLSI.2009.2029117

[46] S. Wei, S. Meguerdichian, and M. Potkonjak, "Malicious circuitry detection using thermal conditioning," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 1136–1145, 2011. [Online]. Available: https://doi.org/10.1109/TIFS.2011.2157341

[47] F. Koushanfar and A. Mirhoseini, "A unified framework for multimodal submodular integrated circuits trojan detection," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 1, pp. 162–174, 2011. [Online]. Available: https://doi.org/10.1109/TIFS.2010.2096811

[48] S. Wei and M. Potkonjak, "Scalable hardware trojan diagnosis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 6, pp. 1049–1057, 2012. [Online]. Available: https://doi.org/10.1109/TVLSI.2011.2147341

[49] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *ACM/IEEE Design Automation Conference*, 2007, pp. 9–14.

[50] J. Li and J. Lach, "At-speed delay characterization for ic authentication and trojan horse detection," in *IEEE International Workshop on Hardware-Oriented Security and Trust*, 2008, pp. 8–14. [Online]. Available: https://doi.org/10.1109/HST.2008.4559038

[51] K. Xiao, X. Zhang, and M. Tehranipoor, "A clock sweeping technique for detecting hardware trojans impacting circuits delay," *IEEE Design Test*, vol. 30, no. 2, pp. 26–34, 2013. [Online]. Available: https://doi.org/10.1109/MDAT.2013.2249555

[52] Y. Jin and Y. Makris, "Hardware trojans in wireless cryptographic ics," *IEEE Design Test of Computers*, vol. 27, no. 1, pp. 26–35, 2010. [Online]. Available: https://doi.org/10.1109/MDT.2010.21

[53] S. Narasimhan, D. Du, R. S. Chakraborty, S. Paul, F. G. Wolff, C. A. Papachristou, K. Roy, and S. Bhunia, "Hardware trojan detection by multiple-parameter side-channel analysis," *IEEE Transactions on Computers*, vol. 62, no. 11, pp. 2183–2195, 2013. [Online]. Available: https://doi.org/10.1109/TC.2012.200

[54] R. S. Chakraborty, F. Wolff, S. Paul, C. Papachristou, and S. Bhunia, "Mero: A statistical approach for hardware trojan detection," in *Cryptographic Hardware and Embedded Systems - CHES 2009*, C. Clavier and K. Gaj, Eds. Springer Berlin Heidelberg, 2009, pp. 396–410.

[55] B. Cakır and S. Malik, "Hardware trojan detection for gate-level ICs using signal correlation based clustering," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, 2015, pp. 471–476.

[56] M. Oya, Y. Shi, M. Yanagisawa, and N. Togawa, "A score-based classification method for identifying hardware-trojans at gate-level netlists," in *Proceedings - Design, Automation and Test in Europe, DATE 2015*, pp. 465–470.

[57] S. Dupuis, G. Di Natale, M.-L. Flottes, and B. Rouzeyre, "Identification of Hardware Trojans triggering signals," in *First Workshop on Trustworthy Manufacturing and Utilization of Secure Devices*, 2013. [Online]. Available: https://hal-lirmm.ccsd.cnrs.fr/lirmm-00991360

[58] H. Salmani, M. Tehranipoor, and J. Plusquellic, "A novel technique for improving hardware trojan detection and reducing trojan activation time," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 1, pp. 112–125, 2012. [Online]. Available: https://doi.org/10.1109/TVLSI.2010.2093547

[59] J. Zhang, F. Yuan, L. Wei, Y. Liu, and Q. Xu, "Veritrust: Verification for hardware trust," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 7, pp. 1148–1161, 2015. [Online]. Available: https://doi.org/10.1109/TCAD.2015.2422836

[60] H. Salmani and M. M. Tehranipoor, "Layout-aware switching activity localization to enhance hardware trojan detection," *IEEE Transactions on Information Forensics and Security*, vol. 7, pp. 76–87, 2012.

[61] K. Xiao, D. Forte, and M. Tehranipoor, "A novel built-in self-authentication technique to prevent inserting hardware trojans," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 12, pp. 1778–1791, 2014. [Online]. Available: https://doi.org/10.1109/TCAD.2014.2356453

[62] B. Khaleghi, A. Ahari, H. Asadi, and S. Bayat-Sarmadi, "Fpga-based protection scheme against hardware trojan horse insertion using dummy logic," *IEEE Embedded Systems Letters*, vol. 7, no. 2, pp. 46–50, 2015. [Online]. Available: https://doi.org/10.1109/LES.2015.2406791

[63] P. Ba, S. Dupuis, M. Palanichamy, M.-L. Flottes, G. Di Natale, and B. Rouzeyre, "Hardware trust through layout filling: A hardware trojan prevention technique," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2016, pp. 254–259. [Online]. Available: https://doi.org/10.1109/ISVLSI.2016.22

[64] Y. Barak, M. Swartz, and Y. Baruch, "O-04 - switching after treatment failure with an SSRI among older depressed inpatients: venlafaxine or a second SSRI?" *European Psychiatry*, vol. 27, p. 1, 2012. [Online]. Available: https://doi.org/10.1016/S0924-9338(12)74104-4

[65] S. Goldwasser and G. N. Rothblum, "On best-possible obfuscation," *Journal of Cryptology*, vol. 27, pp. 480–505, 2013.

[66] M. Xue, J. Wang, Y. Wang, and A. Hu, "Security against hardware trojan attacks through a novel chaos fsm and delay chains array puf based design obfuscation scheme," in *Cloud Computing and Security*, Z. Huang, X. Sun, J. Luo, and J. Wang, Eds. Cham: Springer International Publishing, 2015, pp. 14–24. [Online]. Available: https://doi.org/10.1007/978-3-319-27051-7_2

[67] J. A. Roy, F. Koushanfar, and I. L. Markov, "Epic: Ending piracy of integrated circuits," in *2008 Design, Automation and Test in Europe*, 2008, pp. 1069–1074. [Online]. Available: https://doi.org/10.1109/DATE.2008.4484823

[68] B. Liu and B. Wang, "Reconfiguration-based vlsi design for security," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 5, pp. 98–108, 2015.

[69] J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri, "Fault analysis-based logic encryption," *IEEE Transactions on Computers*, vol. 64, no. 2, pp. 410–424, 2015. [Online]. Available: https://doi.org/10.1109/TC.2013.193

[70] F. Koushanfar, "Provably secure active ic metering techniques for piracy avoidance and digital rights management," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 1, pp. 51–63, 2012. [Online]. Available: https://doi.org/10.1109/TIFS.2011.2163307

[71] J. Zhang, Y. Lin, Y. Lyu, and G. Qu, "A puf-fsm binding scheme for fpga ip protection and pay-per-device licensing," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 6, pp. 1137–1150, 2015. [Online]. Available: https://doi.org/10.1109/TIFS.2015.2400413

[72] L. Li and H. Zhou, "Structural transformation for best-possible obfuscation of sequential circuits," in *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2013, pp. 55–60. [Online]. Available: https://doi.org/10.1109/HST.2013.6581566

[73] K. Vaidyanathan, R. Liu, E. Sumbul, Q. Zhu, F. Franchetti, and L. Pileggi, "Efficient and secure intellectual property (IP) design with split fabrication," in *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, 2014, pp. 13–18. [Online]. Available: https://doi.org/10.1109/HST.2014.6855561

[74] M. Jagasivamani, P. Gadfort, M. Sika, M. Bajura, and M. Fritze, "Split-fabrication obfuscation: Metrics and techniques," *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, pp. 7–12, 2014.

[75] S. Patnaik, M. Ashraf, O. Sinanoglu, and J. Knechtel, "A modern approach to IP protection and trojan prevention: Split manufacturing for 3D ICs and obfuscation of vertical interconnects," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 4, pp. 1815–1834, 2021. [Online]. Available: https://doi.org/10.1109/TETC.2019.2933572

[76] M. Li, B. Yu, Y. Lin, X. Xu, W. Li, and D. Z. Pan, "A practical split manufacturing framework for trojan prevention via simultaneous wire lifting and cell insertion," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2018, pp. 265–270. [Online]. Available: https://doi.org/10.1109/ASPDAC.2018.8297316

[77] S. Patnaik, M. Ashraf, O. Sinanoglu, and J. Knechtel, "Best of both worlds: Integration of split manufacturing and camouflaging into a security-driven CAD flow for 3D ICs," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2018, pp. 1–8. [Online]. Available: https://doi.org/10.1145/3240765.3240784

[78] S. M. H. Shekarian and M. Saheb Zamani, "Improving hardware trojan detection by retiming," *Microprocessors and Microsystems*, vol. 39, no. 3, pp. 145–156, 2015. [Online]. Available: https://doi.org/10.1016/j.micpro.2015.02.002

[79] M. R. Asadi Kouhanjani and A. H. Jahangir, "Improving hardware trojan detection using scan chain based ring oscillators," *Microprocessors and Microsystems*, vol. 63, pp. 55–65, 2018. [Online]. Available: https://doi.org/10.1016/j.micpro.2018.08.008

[80] J. Zhong and J. Wang, "Thermal images based hardware trojan detection through differential temperature matrix," *Optik*, vol. 158, pp. 855–860, 2018. [Online]. Available: https://doi.org/10.1016/j.ijleo.2017.12.145

[81] 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, 2018. [Online]. Available: https://doi.org/10.1016/j.vlsi.2017.11.003

[82] D. McIntyre, F. Wolff, C. Papachristou, S. Bhunia, and D. Weyer, "Dynamic evaluation of hardware trust," in *IEEE International Workshop on Hardware-Oriented Security and Trust*, 2009, pp. 108–111. [Online]. Available: https://doi.org/10.1109/HST.2009.5224990

[83] M. Hicks, M. Finnicum, S. T. King, M. M. K. Martin, and J. M. Smith, "Overcoming an untrusted computing base: Detecting and removing malicious hardware automatically," in *IEEE Symposium on Security and Privacy*, 2010, pp. 159–172. [Online]. Available: https://doi.org/10.1109/SP.2010.18

[84] C. Bao, D. Forte, and A. Srivastava, "Temperature tracking: Toward robust run-time detection of hardware trojans," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1577–1585, 2015. [Online]. Available: https://doi.org/10.1109/TCAD.2015.2424929

[85] J. H. Nisha Jacob, Dominik Merli and G. Sigl, "Hardware trojans: current challenges and approaches," *IET Computer and Digital Technologies*, vol. 8, no. 6, p. 264–273, 2014. [Online]. Available: https://doi.org/10.1049/iet-cdt.2014.0039

[86] H. Martin, P. Peris-Lopez, J. E. Tapiador, E. San Millan, and N. Sklavos, "Hardware trojans in trngs," in *Proccesigns of Trustworthy Manufacturing and Utilization of Secure Devices Workshop, DATE Conference*, 2015.

[87] J. Li, L. Ni, J. Chen, and E. Zhou, "A novel hardware trojan detection based on bp neural network," in *IEEE International Conference on Computer and Communications (ICCC)*, 2016, pp. 2790–2794. [Online]. Available: https://doi.org/10.1109/CompComm.2016.7925206

[88] F. Kounelis, N. Sklavos, and P. Kitsos, "Run-time effect by inserting hardware trojans, in combinational circuits," in *Euromicro Conference on Digital System Design (DSD)*. IEEE, 2017, pp. 287–290.

[89] M. Tehranipoor, "New directions in hardware security," in *International Conference on VLSI Design and International Conference on Embedded Systems (VLSID)*, 2016, pp. 50–52. [Online]. Available: https://doi.org/10.1109/VLSID.2016.149

[90] R. Elnaggar and K. Chakrabarty, "Machine learning for hardware security: Opportunities and risks," *Journal of Electronic Testing*, vol. 34, pp. 183–201, 2018. [Online]. Available: https://doi.org/10.1007/s10836-018-5726-9

[91] Y. Jin, D. Maliuk, and Y. Makris, "Post-deployment trust evaluation in wireless cryptographic ICs," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, 2012, pp. 965–970. [Online]. Available: https://doi.org/10.1109/DATE.2012.6176636

[92] K. G. Liakos, G. K. Georgakilas, S. Moustakidis, P. Karlsson, and F. C. Plessas, "Machine learning for hardware trojan detection: A review," in *Panhellenic Conference on Electronics Telecommunications (PACET)*, 2019, pp. 1–6. [Online]. Available: https://doi.org/10.1109/PACET48583.2019.8956251

[93] J. Li, L. Ni, J. Chen, and E. Zhou, "A novel hardware trojan detection based on BP neural network," in *IEEE International Conference on Computer and Communications*. IEEE, 2016, pp. 2790–2794. [Online]. Available: https://doi.org/10.1109/CompComm.2016.7925206

[94] L. Ni, J. Li, S. Lin, and D. Xin, "A method of noise optimization for hardware trojans detection based on BP neural network," in *IEEE International Conference on Computer and Communications (ICCC)*, 2016, pp. 2800–2804. [Online]. Available: https://doi.org/10.1109/CompComm.2016.7925208

[95] C. H. Kok, C. Y. Ooi, M. Moghbel, N. Ismail, H. S. Choo, and M. Inoue, "Classification of trojan nets based on scoap values using supervised learning," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2019, pp. 1–5. [Online]. Available: https://doi.org/10.1109/ISCAS.2019.8702462

[96] Z. Huang, Q. Wang, Y. Chen, and X. Jiang, "A survey on machine learning against hardware trojan attacks: Recent advances and challenges," *IEEE Access*, vol. 8, pp. 10 796–10 826, 2020. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.2965016

[97] C. Bao, D. Forte, and A. Srivastava, "On application of one-class svm to reverse engineering-based hardware trojan detection," in *Fifteenth International*

Symposium on Quality Electronic Design, 2014, pp. 47–54. [Online]. Available: https://doi.org/10.1109/ISQED.2014.6783305

[98] C. Bao, D. Forte, and B. Srivastava, "On reverse engineering-based hardware trojan detection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 1, pp. 49–57, 2016. [Online]. Available: https://doi.org/10.1109/TCAD.2015.2488495

[99] A. N. Abdurrahman and Z. A. Mohamed, "An efficient reverse engineering hardware trojan detector using histogram of oriented gradients," *J Electron Test*, vol. 35, no. 33, p. 93–105, 2017. [Online]. Available: https://doi.org/10.1007/s10836-016-5631-z

[100] A. Kulkarni, Y. Pino, M. French, and T. Mohsenin, "Real-time anomaly detection framework for many-core router through machine-learning techniques," *J. Emerg. Technol. Comput. Syst.*, vol. 13, no. 1, jun 2016. [Online]. Available: https://doi.org/10.1145/2827699

[101] A. Kulkarni, Y. Pino, and T. Mohsenin, "Svm-based real-time hardware trojan detection for many-core platform," in *International Symposium on Quality Electronic Design (ISQED)*, 2016, pp. 362–367. [Online]. Available: https://doi.org/10.1109/ISQED.2016.7479228

[102] A. Kulkarni, T. Mohsenin, and Y. Pino, "Adaptive real-time trojan detection framework through machine learning," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2016, pp. 120–123. [Online]. Available: https://doi.org/10.1109/HST.2016.7495568

[103] S. Faezi, R. Yasaei, A. Barua, and M. A. A. Faruque, "Brain-inspired golden chip free hardware trojan detection," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 2697–2708, 2021. [Online]. Available: https://doi.org/10.1109/TIFS.2021.3062989

[104] D. Jap, W. He, and S. Bhasin, "Supervised and unsupervised machine learning for side-channel based trojan detection," *2016 IEEE 27th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*, pp. 17–24, 2016.

[105] M. Xue, J. Wang, and A. Hu, "An enhanced classification-based golden chips-free hardware trojan detection technique," in *IEEE Asian Hardware-Oriented Security and Trust (AsianHOST)*, 2016, pp. 1–6. [Online]. Available: https://doi.org/10.1109/AsianHOST.2016.7835553

[106] F. K. Lodhi, S. R. Hasan, O. Hasan, and F. Awwadl, "Power profiling of microcontroller's instruction set for runtime hardware trojans detection without golden circuit models," in *Design, Automation Test in Europe Conference Exhibition (DATE)*, 2017, pp. 294–297. [Online]. Available: https://doi.org/10.23919/DATE.2017.7927002

[107] H. Salmani, "Cotd: Reference-free hardware trojan detection and recovery based on controllability and observability in gate-level netlist," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 2, pp. 338–350, 2017. [Online]. Available: https://doi.org/10.1109/TIFS.2016.2613842

[108] N. Shang, A. Wang, Y. Ding, K. Gai, L. Zhu, and G. Zhang, "A machine learning based golden-free detection method for command-activated hardware trojan," *Information Sciences*, vol. 540, pp. 292–307, 2020. [Online]. Available: https://doi.org/10.1016/j.ins.2020.05.053

[109] K. Hasegawa, M. Oya, M. Yanagisawa, and N. Togawa, "Hardware trojans classification for gate-level netlists based on machine learning," in *IEEE International Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2016, pp. 203–206. [Online]. Available: https://doi.org/10.1109/IOLTS.2016.7604700

[110] K. Hasegawa, M. Yanagisawa, and N. Togawa, "Trojan-feature extraction at gate-level netlists and its application to hardware-trojan detection using random forest classifier," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2017, pp. 1–4. [Online]. Available: https://doi.org/10.1109/ISCAS.2017.8050827

[111] T. Inoue, K. Hasegawa, M. Yanagisawa, and N. Togawa, "Designing hardware trojans and their detection based on a svm-based approach," in *International Conference on ASIC (ASICON)*, 2017, pp. 811–814. [Online]. Available: https://doi.org/10.1109/ASICON.2017.8252600

[112] K. Hasegawa, M. Yanagisawa, and N. Togawa, "A hardware-trojan classification method utilizing boundary net structures," in *International Conference on Consumer Electronics, (ICCE)*, 2018, pp. 1–4. [Online]. Available: https://doi.org/10.1109/ICCE.2018.8326247

[113] T. Iwase, Y. Nozaki, M. Yoshikawa, and T. Kumaki, "Detection technique for hardware trojans using machine learning in frequency domain," in *IEEE Global Conference on Consumer Electronics (GCCE)*, 2015, pp. 185–186. [Online]. Available: https://doi.org/10.1109/GCCE.2015.7398569

[114] S. Wang, X. Dong, K. Sun, Q. Cui, D. Li, and C. He, "Hardware trojan detection based on ELM neural network," in *IEEE International Conference on Computer Communication and the Internet (ICCCI)*, 2016, pp. 400–403.

[115] F. Lodhi, I. Abbasi, F. Khalid, O. Hasan, F. Awwad, and S. R. Hasan, "A self-learning framework to detect the intruded integrated circuits," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2016, pp. 1702–1705. [Online]. Available: https://doi.org/10.1109/ISCAS.2016.7538895

[116] N. Noor, N. Sjarif, N. Azmi, S. Daud, and K. Kamardin, "Hardware trojan identification using machine learning-based classification," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 9, no. 3-4, pp. 23–27, 2017.

[117] S. Shahparnia and O. Ramahi, "Electromagnetic interference (emi) reduction from printed circuit boards (PCB) using electromagnetic bandgap structures," *IEEE Transactions on Electromagnetic Compatibility*, vol. 46, no. 4, pp. 580–587, 2004. [Online]. Available: https://doi.org/10.1109/TEMC.2004.837671

[118] P. Johnston, "Printed circuit board design guidelines for ball grid array packages," *Journal Surface Mount Technology*, vol. 9, pp. 12–18, 1996.

[119] "Requirements for electrical testing of unpopulated printed boards, 9252B, bannockburn IL," 2016.

[120] "IPC, generic standard on printed board design, 2221b, Bannockburn, IL," 2012.

[121] F. Koushanfar and M. Tehranipoor, "A survey of hardware trojan taxonomy and detection," *IEEE Design Test of Computers*, vol. 27, no. 01, pp. 10–25, 2010. [Online]. Available: https://doi.org/10.1109/MDT.2010.7

[122] M. Azhagan, D. Mehta, H. Lu, and S. Agrawal, "A review on automatic bill of material generation and visual inspection on PCBs," in *International Symposium for Testing and Failure Analysis*, 2019, pp. 256–265. [Online]. Available: https://doi.org/10.31399/asm.cp.istfa2019p0256

[123] P. Malge and R. Nadaf, "PCB defect detection, classification and localization using mathematical morphology and image processing tools," *International Journal of Computer Applications*, vol. 87, no. 9, pp. 40–45, 2014.

[124] S. Tang, F. He, X. Huang, and J. Yang, "Online PCB defect detector on a new PCB defect dataset," 2019. [Online]. Available: https://doi.org/10.48550/ARXIV.1902.06197

[125] H. Wu, X. Zhang, and S. Hong, "A visual inspection system for surface mounted components based on color features," in *International Conference on Information and Automation*, 2009, pp. 571–576. [Online]. Available: https://doi.org/10.1109/ICINFA.2009.5204988

[126] A. Crispin and V. Rankov, "Automated inspection of PCB components using a genetic algorithm template-matching approach," *Int Journal Adv Manuf Technol*, vol. 35, p. 293–300, 2007. [Online]. Available: https://doi.org/10.1007/s00170-006-0730-0

[127] C. Szymanski and M. R. Stemmer, "Automated PCB inspection in small series production based on sift algorithm," in *IEEE International Symposium on Industrial Electronics (ISIE)*, 2015, pp. 594–599. [Online]. Available: https://doi.org/10.1109/ISIE.2015.7281535

[128] R. Smith, "An overview of the tesseract OCR engine," in *International Conference on Document Analysis and Recognition (ICDAR)*, vol. 2, 2007, pp. 629–633. [Online]. Available: https://doi.org/10.1109/ICDAR.2007.4376991

[129] F. Zhang, A. Hennessy, and S. Bhunia, "Robust counterfeit PCB detection exploiting intrinsic trace impedance variations," *IEEE VLSI Test Symposium (VTS)*, pp. 1–6, 2015.

[130] N. Ahmed, M. Tehranipoor, and V. Jayaram, "Timing-based delay test for screening small delay defects," in *Proceedings of Design Automation Conference*, ser. DAC '06, 2006, p. 320–325. [Online]. Available: https://doi.org/10.1145/1146909.1146993

[131] K. Cheng, "Transition fault testing for sequential circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 12, pp. 1971–1983, 1993. [Online]. Available: https://doi.org/10.1109/43.251160

[132] J. Hamlet, M. Martin, and N. Edwards, "Unique signatures from printed circuit board design patterns and surface mount passives," in *International Carnahan Conference on Security Technology (ICCST)*, 2017, pp. 1–6. [Online]. Available: https://doi.org/10.1109/CCST.2017.8167796

[133] S. Quadir, J. Chen, D. Forte, N. Asadizanjani, S. Shahbazmohamadi, L. Wang, J. Chandy, and M. Tehranipoor, "A survey on chip to system reverse engineering," *J. Emerg. Technol. Comput. Syst.*, vol. 13, no. 1, 2016. [Online]. Available: https://doi.org/10.1145/2755563

[134] Z. Guo, M. Tehranipoor, D. Forte, and J. Di, "Investigation of obfuscation-based anti-reverse engineering for printed circuit boards," in *ACM/EDAC/IEEE Design Automation Conference (DAC)*, 2015, pp. 1–6. [Online]. Available: https://doi.org/10.1145/2744769.2744862

[135] Z. Guo, J. Di, M. M. Tehranipoor, and D. Forte, "Obfuscation-based protection framework against printed circuit boards unauthorized operation and reverse engineering," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 22, no. 3, 2017. [Online]. Available: https://doi.org/10.1145/3035482

[136] Z. Guo, S. Chowdhury, M. Tehranipoor, and D. Forte, "Permutation network de-obfuscation: A delay-based attack and countermeasure investigation," *J. Emerg. Technol. Comput. Syst.*, vol. 16, no. 2, 2020. [Online]. Available: https://doi.org/10.1145/3371407

[137] M. Nishizawa, K. Hasegawa, and N. Togawa, "Capacitance measurement of running hardware devices and its application to malicious modification detection," in *IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, 2018, pp. 362–365. [Online]. Available: https://doi.org/10.1109/APCCAS.2018.8605668

[138] Z. Guo, X. Xu, M. Tehranipoor, and D. Forte, "MPA: Model-assisted PCB attestation via board-level RO and temperature compensation," in *Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, 2017, pp. 25–30. [Online]. Available: https://dx.doi.org/10.1109/AsianHOST.2017.8353990

[139] D. Zhang, Y. Han, and Q. Ren, "A novel authorization methodology to prevent counterfeit PCB/equipment through supply chain," in *IEEE International Conference on Integrated Circuits and Microsystems (ICICM)*, 2019, pp. 128–132. [Online]. Available: https://dx.doi.org/10.1109/ICICM48536.2019.8977190

[140] X. Wang, Y. Han, and M. Tehranipoor, "System-level counterfeit detection using on-chip ring oscillator array," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 12, pp. 2884–2896, 2019. [Online]. Available: https://dx.doi.org/10.1109/TVLSI.2019.2930532

[141] G. Piliposyan, S. Khursheed, and D. Rossi, "Hardware trojan detection on a PCB through differential power monitoring," *IEEE Transactions on Emerging Topics in Computing*, 2022. [Online]. Available: https://dx.doi.org/10.1109/TETC.2020.3035521

[142] Z. Guo, X. Xu, M. Tehranipoor, and D. Forte, "EOP: An encryption-obfuscation solution for protecting PCBs against tampering and reverse engineering," 2019. [Online]. Available: https://arxiv.org/abs/1904.09516

[143] N. Asadizanjani, M. Tehranipoor, and D. Forte, "Non-destructive PCB reverse engineering using x-ray micro computed tomography," in *IEEE Transactions on Components, Packaging, and Manufacturing Technology*, 2017, pp. 292–299.

[144] X. Zhang, K. Xiao, M. Tehranipoor, J. Rajendran, and R. Karri, "A study on the effectiveness of trojan detection techniques using a red team blue team approach," in *IEEE VLSI Test Symposium (VTS)*, 2013, pp. 1–3. [Online]. Available: https://10.1109/VTS.2013.6548922

[145] X. Wang, H. Salmani, M. Tehranipoor, and J. Plusquellic, "Hardware trojan detection and isolation using current integration and localized current analysis," in *IEEE international symposium on defect and fault tolerance of VLSI systems*. IEEE, 2008, pp. 87–95. [Online]. Available: https://10.1109/DFT.2008.61

[146] S. M. Ross, *Hardware Trojans and Piracy of PCBs*. Elsevier, 2014, p. 624.

[147] M. Dhwani, H. Lu, O. Paradis, A. Mukhil, T. Rahman, Y. Iskander, P. Chawla, D. Woodard, M. Tehranipoor, and N. Asadizanjani, "The Big Hack Explained: Detection and prevention of PCB supply chain implants," *J. Emerg. Technol. Comput. Syst.*, vol. 16, no. 4, 2020. [Online]. Available: https://doi.org/10.1145/3401980

[148] M. Moganti, F. Ercal, C. H. Dagli, and S. Tsunekawa, "Automatic PCB inspection algorithms: A survey," *Computer Vision and Image Understanding*, vol. 63, no. 2, pp. 287–313, 1996. [Online]. Available: https://doi.org/10.1006/cviu.1996.0020

[149] W. Wang, S. Chen, L. Chen, and W. Chang, "A machine vision based automatic optical inspection system for measuring drilling quality of printed circuit boards," *IEEE Access*, vol. 5, pp. 10 817–10 833, 2017. [Online]. Available: https://10.1109/ACCESS.2016.2631658

[150] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Computer Vision- ECCV*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 404–417.

[151] H. Pearce, V. R. Surabhi, P. Krishnamurthy, J. Trujillo, R. Karri, and F. Khorrami, "Detecting hardware trojans in PCBs using side channel loopbacks," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 30, no. 7, 7 2022. [Online]. Available: https://doi.org/10.1109/tvlsi.2022.3171174

[152] T. Mosavirik, P. Schaumont, and S. Tajik, "Impedanceverif: On-chip impedance sensing for system-level tampering detection," Cryptology ePrint Archive, Paper 2022/946, 2022, https://eprint.iacr.org/2022/946. [Online]. Available: https://eprint.iacr.org/2022/946

[153] T. Mosavirik, F. Ganji, P. Schaumont, and S. Tajik, "Scatterverif: Verification of electronic boards using reflection response of power distribution network," *J. Emerg. Technol. Comput. Syst.*, vol. 18, no. 4, 2022. [Online]. Available: https://doi.org/10.1145/3513087

[154] H. Zhu, H. Shan, D. Sullivan, X. Guo, Y. Jin, and X. Zhang, "Pdnpulse: Sensing anomaly with the intrinsic power delivery network," 2022. [Online]. Available: https://arxiv.org/abs/2204.02482

[155] D. Fujimoto, S. Nin, Y. Hayashi, N. Miura, M. Nagata, and T. Matsumoto, "A demonstration of a HT-detection method based on impedance measurements of the wiring around ICs," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, no. 10, pp. 1320–1324, 2018.

[156] F. Pedregosa, G. Varoquaux, A. Gramfort *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[157] J. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, p. 293–300, Jun. 1999. [Online]. Available: https://doi.org/10.1023/A:1018628609742

[158] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001. [Online]. Available: https://doi.org/10.1162/089976601750264965

[159] B. Schölkopf, R. Williamson, A. Smola, J. Shawe-Taylor, J. Platt *et al.*, "Support vector method for novelty detection." in *NIPS*, vol. 12. Citeseer, 1999, pp. 582–588.

[160] B. Schölkopf, A. Smola, F. Bach *et al.*, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[161] M. Breunig, H. Kriegel, R. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2000, p. 93–104. [Online]. Available: https://doi.org/10.1145/342009.335388

[162] C. Dong, Y. Liu, J. Chen, X. Liu, W. Guo, and Y. Chen, "An unsupervised detection approach for hardware trojans," *IEEE Access*, vol. 8, pp. 158 169–158 183, 2020. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.3001239

[163] R. Domingues, M. Filippone, P. Michiardi, and J. Zouaoui, "A comparative evaluation of outlier detection algorithms: Experiments and analyses," *Pattern Recognition*, vol. 74, pp. 406–421, 2018. [Online]. Available: https://doi.org/10.1016/j.patcog.2017.09.037

[164] S. Maji, A. Berg, and J. Malik, "Efficient classification for additive kernel SVMs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 66–77, 2012.

[165] I. Tsang, J. Kwok, P. Cheung, and N. Cristianini, "Core vector machines: Fast SVM training on very large data sets." *Journal of Machine Learning Research*, vol. 6, no. 4, 2005.

[166] M. Claesen, F. Smet, J. Suykens, and B. Moor, "Fast prediction with SVM models containing RBF kernels," *arXiv preprint arXiv:1403.0736*, 2014.

[167] E. Alpaydin, *Introduction to machine learning*. MIT press, 2020.

[168] Z. Ghafoori, S. Erfani, S. Rajasegarar, J. Bezdek, S. Karunasekera, and C. Leckie, "Efficient unsupervised parameter estimation for one-class support vector machines," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 10, pp. 5057–5070, 2018. [Online]. Available: https://10.1109/TNNLS.2017.2785792

[169] K. Divya and N. S. Kumaran, "Improved outlier detection using classic KNN algorithm," in *International Journal of Engineering and Technology (IRJET)*, vol. 3, no. 12, 2016, pp. 892–898.

[170] G. Piliposyan and S. Khursheed, "Computer vision for hardware trojan detection on a PCB using siamese neural network," in *IEEE Physical Assurance and Inspection of Electronics (PAINE)*, 2022, pp. 1–7.

[171] H. Lu, D. Capecci, D. Pallabi Ghosh, and D. Woodard, *Computer vision for hardware security*. Springer, 2021, ch. 18, pp. 493–527. [Online]. Available: https://doi.org/10.1007/978-3-030-64448-2_18

[172] G. Acciani, G. Brunetti, and G. Fornarelli, "Application of neural networks in optical inspection and classification of solder joints in surface mount technology," *IEEE Transactions on industrial informatics*, vol. 2, no. 3, pp. 200–209, 2006. [Online]. Available: https://doi.org/10.1109/TII.2006.877265

[173] J. Grzyb, K. Statnikov, A. Hadi, and U. Pfeiffer, "All-silicon integrated THz harmonic source and receiver components for future active imaging modalities," in *International Conference on Infrared, Millimeter, and Terahertz Waves (IRMMW-THz)*, 2014, pp. 1–2. [Online]. Available: https://doi.org/10.1109/IRMMW-THz.2014.6956429

[174] L. Watkins, "Inspection of integrated circuit photomasks with intensity spatial filters," *Proceedings of the IEEE*, vol. 57, no. 9, pp. 1634–1639, 1969. [Online]. Available: https://doi.org/10.1109/PROC.1969.7348

[175] F. Xie, A. Uitdenbogerd, and A. Song, "Detecting PCB component placement defects by genetic programming," in *IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 1138–1145.

[176] K. Sundaraj, "PCB inspection for missing or misaligned components using background subtraction," *WSEAS Transactions on Information Science and Applications archive*, vol. 6, pp. 778–787, 2009.

[177] H. Cho and T. Park, "Wavelet transform based image template matching for automatic component inspection," *Int Journal Adv Manuf Technol*, vol. 50, p. 1033–1039, 2010. [Online]. Available: https://doi.org/10.1007/s00170-010-2567-9

[178] O. P. Paradis, N. T. Jessurun, M. Tehranipoor, and N. Asadizanjani, "Color normalization for robust automatic bill of materials generation and visual inspection of PCBs," in *ISTFA*, 2020, pp. 172–179. [Online]. Available: https://doi.org/10.31399/asm.cp.istfa2020p0172

[179] W. Zhao, S. Gurudu, S. Taheri, S. Ghosh, M. Sathiaseelan, A. Mukhil, and N. Asadizanjani, "PCB component detection using computer vision for hardware assurance," *Big Data and Cognitive Computing*, vol. 6, no. 2, p. 39, 2022. [Online]. Available: https://doi.org/10.3390/bdcc6020039

[180] S. Youn, Y. Lee, and T. Park, "Automatic classification of smd packages using neural network," in *IEEE/SICE International Symposium on System Integration*. IEEE, 2014, pp. 790–795. [Online]. Available: https://doi.org/10.1109/SII.2014.7028139

[181] D. Lim, Y. Kim, and T. Park, "SMD classification for automated optical inspection machine using convolution neural network," in *IEEE International Conference on Robotic Computing (IRC)*. IEEE, 2019, pp. 395–398. [Online]. Available: https://doi.org/10.1109/IRC.2019.00072

[182] M. A. Reza, Z. Chen, and D. J. Crandall, "Deep neural network–based detection and verification of microelectronic images," *Journal of Hardware and Systems Security*, vol. 4, no. 1, pp. 44–54, 2020.

[183] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," *Advances in neural information processing systems*, vol. 6, pp. 737–744, 1993.

[184] G. Koch, R. Zemel, R. Salakhutdinov *et al.*, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2, 2015.

[185] D. David and A. NG, "Designing effective traditional and deep learning-based inspection systems," *Vision Systems Desugn*, vol. 26, no. 5, pp. 10–14, 2021.

[186] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[187] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, p. 381–395, 1981. [Online]. Available: https://doi.org/10.1145/358669.358692

[188] LinkedAi, "Flip," https://github.com/LinkedAi/flip.

[189] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[190] L. Martínez, S. Khursheed, T. Alnuayri, and D. Rossi, "Online remaining useful lifetime prediction using support vector regression," *IEEE Transactions on Emerging Topics in Computing*, vol. 10, no. 3, pp. 1546–1557, 2022.