Defending Against Hardware Trojans

Contents

Chapter 1

ABSTRACT

Hardware Trojans are a growing concern in modern computer systems, and protecting against them is a critical challenge for hardware designers and system architects. Physical unclonable functions (PUFs) are a hardware security mechanism that leverage the unique physical properties of a device to generate cryptographic keys, making it challenging for attackers to tamper with or clone the device. This study provides an overview of the problem of hardware Trojans and investigates the role of PUFs in defending against them. It discusses various types of PUFs, including SRAM, ring oscillator, and delay-based PUFs, and examines recent research on PUF-based solutions for hardware Trojan detection, such as machine learning and error-correcting codes. The study also highlights the challenges and limitations of using PUFs for hardware security and concludes that PUFs offer a promising and effective approach to defending against hardware Trojans. However, further research and development are necessary to improve their reliability and practicality. The primary objective of this study is to provide a comprehensive understanding of the role of PUFs in hardware security and to inform future efforts in this area.

Chapter 2

INTRODUCTION

The integration of technology in daily life has led to unprecedented convenience and efficiency. However, it has also resulted in new challenges and threats that require innovative solutions to mitigate the risks. Hardware Trojans are one such challenge, which involves malicious modifications or insertions in the hardware design, compromising the security and functionality of the system. The proliferation of hardware Trojans has become a major concern for various industries, including defense, aerospace, and critical infrastructure, as they can cause catastrophic failures, disrupt operations, and steal sensitive information.

To defend against these threats, researchers and practitioners have been exploring various approaches, including the use of Physical Unclonable Functions (PUFs). PUFs are unique and inherent properties of physical systems that can be used for authentication, identification, and key generation. They offer a secure and reliable way to protect against hardware Trojans, as they are resistant to duplication, reverse-engineering, and tampering. The effectiveness of PUFs in various applications has been demonstrated in numerous studies.

This report aims to explore the role of PUFs in defending against hardware Trojans. It provides an overview of the concept, principle of working, technology considerations, advantages, and limitations of PUFs. Additionally, the report discusses various real-world applications of PUFs and highlights their strengths and weaknesses. Recommendations for future research directions and potential improvements in PUF technology are also provided in the conclusion.

Hardware Trojans have been discovered historically in various systems, such as microprocessors, FPGAs, and ASICs. They can be inserted during the design phase or manufacturing process, and can be triggered by specific inputs or environmental conditions. Detecting and preventing hardware Trojans is complex and challenging, as they can be designed to be stealthy and hard to detect.

PUFs provide a promising solution to this problem by utilizing the physical properties of the hardware to provide unique and unpredictable responses to specific inputs. They can verify the authenticity of the hardware and detect any modifications or alterations. PUFs can also be used for key generation, which can be used for encryption and authentication purposes. Overall, PUFs have the potential to provide a robust and reliable defense against hardware Trojans. This report explores the various aspects of PUF technology and its applications in defending against hardware Trojans, including a literature review of previous work done in this field.

Chapter 3

HARDWARE TROJANS

Hardware Trojans are malicious modifications that are inserted into a system during the design or manufacturing stage. Once triggered, these Trojans can perform various functions such as data leakage, denial of service attacks, or remote control. HTs can be inserted at any stage of the design and manufacturing process, making it challenging to detect them. The damage caused by HTs can be catastrophic, especially in critical systems such as aerospace, defense, and healthcare. Thus, there is a need to develop techniques that can detect and defend against these attacks.

Physical attributes of HTs can be classified based on their size, type, distribution, and structure. Types of HTs include functional and parametric. Functional HTs are introduced through the addition or removal of gates or transistors, while parametric HTs are introduced by modifying existing wires or logic. The distribution of HTs can be tight or loose, depending on how the components are topologically placed in the chip. The structure of HTs is designed to avoid detection by not changing the physical layout of the circuit.

HTs can be activated by internal or external triggers. Some HTs do not require a trigger to be activated and can disrupt the system's function anytime. Trigger HTs are activated by a deterministic event, such as sensing the environment or conditions around the device. HTs can have digital or analog payloads, which modify logic values at specific internal payload nodes or affect parameters of the circuit such as performance, power, noise, etc.

Physical Unclonable Functions (PUFs) are a promising technique for defending against HTs. PUFs use the unique physical characteristics of a chip to generate a unique identification code that is difficult to replicate. These physical characteristics can be small variations in the manufacturing process, such as random dopant fluctuations or variations in the shape and size of transistors. PUFs are integrated into the design of the chip and can be used to authenticate the chip and detect any modifications made to it during the manufacturing process.

Hardware Trojan can introduce vulnerabilities or alter the functionality of a design in a way that may go unnoticed. They can be introduced at various stages of the design process, including fabrication, testing, packaging, and deployment. The impact of hardware Trojans can be significant, and certain applications are more likely to be targeted by attackers using hardware Trojans, such as military and aerospace applications, financial or transportation systems, IoT devices, and commercial devices.

To prevent hardware Trojans, designers should implement rigorous security measures throughout the design process. These measures may include using trusted third-party vendors, performing regular security audits, and implementing hardware security modules. By taking proactive steps to prevent hardware Trojans, designers can help ensure the integrity and reliability of their hardware designs.

3.1 The Threat

Hardware Trojan attacks have emerged as a major security concern for integrated circuits (ICs). These attacks relate to malicious modifications of an IC during design or fabrication in an untrusted design house or foundry, which involve untrusted people, design tools, or components. Such modifications can give rise to undesired functional behavior of an IC, or provide covert channels or backdoor through which sensitive information can be leaked. An adversary is expected to make a Trojan stealthy in nature that evades detection through conventional postmanufacturing test, but manifests during long hours of field operation. For an IC with moderate complexity, the number of possible Trojans can be inordinately large with varying activation mechanisms (referred to as triggers) and effects (referred to as payloads). Figure 3.1 shows a simplified block diagram of a hardware Trojan, which causes a malfunction (by modifying signal S to S'), when triggered i.e., when the activation condition realized by the trigger logic is true. Such malicious inclusions effectively act as "spies or terrorists" on chip and can be extremely powerful, potentially leading to catastrophic consequences in diverse applications.

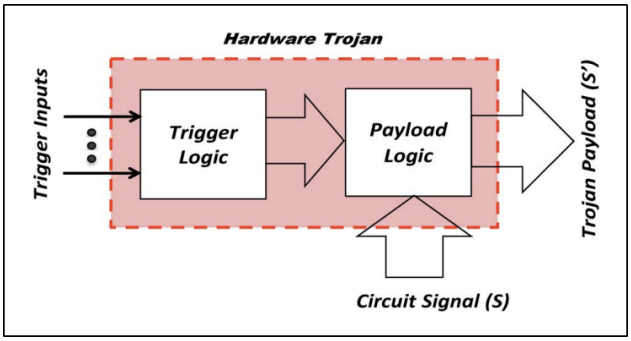

Figure 3.1: General structure of a hardware Trojan in a design.

These malicious circuits have been popularly referred to as "hardware Trojans" similar to the

software Trojans, which attack the operating system (OS) of a computer. The nomenclature is derived from a mythological incident attributed to the ancient Greeks in the Trojan war, where a wooden horse was gifted to the Trojan army who took it into their city walls without realizing that the enemy (Greek) soldiers were hidden inside the hollow horse. The seemingly trustworthy horse conditionally turned into a powerful and malicious weapon that drastically affected the course of the Trojan war. Similar to its mythical analogy, the two main features of a hardware Trojan are as follows: **1)** it should have a malicious intent; and **2)** it should evade detection under conventional postmanufacturing test/validation process.

Current economic trend plays a major role in enhancing the vulnerability to Trojan attacks. IC design and manufacturing practices increasingly rely on untrusted parties and entities in the IC life cycle. Economic factors dictate that most of the modern ICs are manufactured in unsecured fabrication facilities. Moreover, modern IC design often involves intellectual property (IP) cores supplied by untrusted third-party vendors, outsourced design and test services, as well as electronic design automation (EDA) software tools supplied by different vendors. Such a business model has, to a large extent, relinquished the control that IC design houses had over the design and manufacture of ICs making them vulnerable to malicious implants. Figure 3.2 illustrates different steps of a typical IC life cycle and the possibility of Trojan attacks in these steps. Each party associated with the design and fabrication of an IC can be a potential adversary who can tamper it. Such tampering can be accomplished through add/delete/alteration of circuit structure or through modification of manufacturing process steps that causes reliability issues in ICs. From an attacker's perspective, the objective of such attacks can be manifold, e.g., to malign the image of a company to gain competitive edge in the market; disrupt major national infrastructure by causing malfunction in electronics used in mission-critical systems; or leak secret information from inside a chip to illegally access a secure system. Concerns about this vulnerability and the resultant compromise of system security have been expressed globally, especially since recent discoveries point to feasibility of such attacks. Moreover, several unexplained military mishaps in the past have been attributed to the presence of malicious hardware modifications. Recent investigations have shown that an intelligent adversary can mount a hard-to-detect Trojan attack using just a few transistors or logic gates in a large multimillion transistor system-on-chip (SoC) design, or by selectively changing specific process steps, e.g., the doping profile, to affect the operational reliability of a circuit.

Ideally, any undesired modification made to an IC should be detectable by pre-silicon verification/simulation or post-silicon testing. However, pre-silicon verification or simulation requires a completely specified golden model of the entire IC. This might not be always available, especially for IP-based designs where IPs can come from third-party vendors. Besides, a large multimodule design is usually not amenable to exhaustive verification. Post-silicon, the design can be verified either through destructive depackaging and reverse engineering of the IC, or by comparing its functionality or circuit characteristics with a golden version of the IC. However, state-of-the-art approaches do not allow destructive verification of ICs to be either cost effective or scalable. Moreover, as pointed out in, it is possible for the adversary to insert Trojans in only

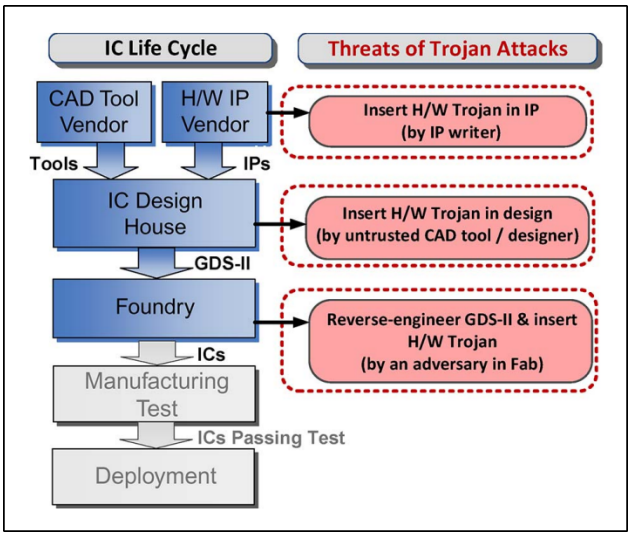

Figure 3.2: Hardware Trojan attacks by different parties at different stages of IC life cycle.

some ICs on a wafer, not the entire population, which limits the usefulness of a destructive approach. Traditional postmanufacturing testing is not suitable for detecting hardware Trojans. This is due to the stealthy nature of hardware Trojans and the vast spectrum of possible Trojan instances an adversary can exploit. Hardware Trojan attacks raise a new set of challenges for trusted operation of electronics in field. It demands trust validation of ICs with respect to malicious design modification at various stages of the IC lifecycle, where untrusted components/personnel are involved. In particular, it introduces the requirement for reliable detection of any malicious design modification during postmanufacturing test. It also imposes a requirement for trust validation in hardware IP cores obtained from untrusted third-party vendors.

3.2 Trojan Versus Fault

Postmanufacturing test, both structural and functional, is targeted to detect different types of manufacturing defects. Faults such as stuck-at-faults or path delay faults are logical models of physical defects (e.g., resistive short or open). Table below compares the properties of hardware Trojans with those of faults. Unlike faults in a design, which occur due to imperfections introduced during the manufacturing process, hardware Trojans are deliberately inserted by an adversary to serve a specific malicious purpose. While a fault is activated at a known functional state of a circuit (e.g., a stuck-at-0 fault at the output of an inverter can be activated by sensitizing its input to a logical value of 0), a Trojan can be designed to activate at an arbitrary complex condition, including a sequence of events at internal circuit nodes.

	Fault	Hardware Trojan
Activation	Usually at known functional state	Arbitrary combination/sequence of internal circuit states (digital/analog)
Insertion Agent	*Accidental* (due to imperfection in manufacturing process)	*Intentional* (inserted by an adversary during IC design or fabrication)
Manifestation	Functional/parametric failure	Functional/parametric failure or information leakage

Figure 3.3: Comparison Between Faults and Hardware Trojan Attacks

3.3 Software Versus Hardware Trojans

Below table distinguishes hardware Trojans from its software counterparts in terms of key properties. A software Trojan horse (STH) is a type of malware program with malicious code that gains privileged access to the OS and may steal information or cause harm to the host computer (e.g., erase or corrupt data). An STH attack can often be cured in the field by running an antiTrojan program that monitors and removes the Trojan. Hardware Trojans are inserted into an IC before it is fabricated. Unlike an STH, a hardware Trojan is virtually impossible to remove from a chip after fabrication and hence can be extremely difficult to remedy during field operation.

	Software Trojan	Hardware Trojan
Activation	A type of malware that resides in a code and activates during its execution	Resides in hardware (e.g. IC) and activates during its operation
Infection	Spreads through user interaction e.g. downloading and running a file from Internet	Inserted through untrusted entities in design or fabrication house
Remedy	Can be *removed in field* through S/W support	*Cannot be removed* once IC is fabricated

Figure 3.4: Comparison Between Software and Hardware Trojans

3.4 Trojan Attacks Through Hardware IP or CAD Tools

SoC design based on reusable hardware IP is now a pervasive practice in the semiconductor industry due to the dramatic reduction in design/verification cost and time it offers. This growing reliance on reusable preverified hardware IPs and a wide array of design automation tools during SoC designVoften gathered from untrusted thirdparty vendorsVseverely affects the security and trustworthiness of SoC computing platforms. The possibility of Trojan attacks in third-party IP poses a major integrity concern to SoC designers. Verification of trust of an IP acquired from untrusted third-party sources can be extremely challenging due to lack of golden models. Conventional verification approaches, which rely on the presence of a golden or reference design, do not work in case of third-party IPs. On the other hand, functional simulation or emulation does not provide adequate coverage due to often incomplete functional specifications. Thus, even though simulation can validate the correctness of a design with respect to functional

specifications, it cannot provide assurance against additional functionality due to a Trojan. For example, if a processor IP core triggers a malicious memory write for an "add" instruction with a specific rare combination of operand values, both simulation and emulation are very likely to fail, since they may not excite all rare conditions. Similarly, untrusted computer-aided design (CAD) tools can potentially cause malicious inclusions in a design. Such attacks can be introduced early in the design cycle, so that they can evade subsequent verification steps. Often, a designer uses a CAD tool suite from the same vendor. In that case, a verification tool from a vendor can potentially ignore a malicious insertion by the synthesis engine from the same vendor.

3.5 Complex Attack Mode

Malicious collusion between multiple parties at different stages of the design, manufacturing, and deployment can make a hardware Trojan attack more potent. One such example is presented in [19], where the inserted Trojan circuitry leaks information through a covert side channel that allows conspiring malicious parties to discover the encryption key. Another example is presented in [20], where focused ion beams help insert a dormant Trojan to a functional unit in the fabrication facility. A general model of such an attack, referred to as multilevel attack. Using a crypto module as a case study, it analytically shows that the resultant attack poses a significantly stronger threat than that from a single adversary. Such attacks can easily bypass both pre-silicon verification as well as postmanufacturing test. An example would be a fault attack in an advanced encryption system (AES) module, which enables only the malicious parties, who are part of the conspiracy, to retrieve the cipher key. The Trojan could also be a passive hardware entity, which simply serves to help malicious software circumvent the hardware protection mechanisms. Moreover, a Trojan circuit can be localized or distributed in a chip, and a trigger condition or the payload of a Trojan can be digital or analog, e.g., a Trojan can be triggered by changes in temperature. Becker et al. have demonstrated that by selectively changing the dopant level in the transistors, they are able to create Trojans that are resistant against existing detection techniques. These Trojans are nonintrusive and difficult to detect since they do not alter any functionality and are not optically observable.

3.6 Trojan Model, Instances, and Classification

3.6.1 Trojan Models and Examples

An intelligent adversary is expected to hide such tampering with an IC's behavior in a way that makes it extremely difficult to detect with conventional postmanufacturing testing. Intuitively, it means that the adversary would ensure that such tampering is manifested or triggered under very rare conditions at the internal nodes, which are unlikely to arise during testing but can occur during long hours of field operation. Figure 3.5 shows some examples of different types of hardware Trojans. The combinational Trojan, as shown in Figure 3.5(a), does not contain any state element (e.g., flip-flop or latch) and depends only on the simultaneous occurrence of a set of rare node conditions (e.g., a predefined value on node sets a and b) to trigger a malfunction (by

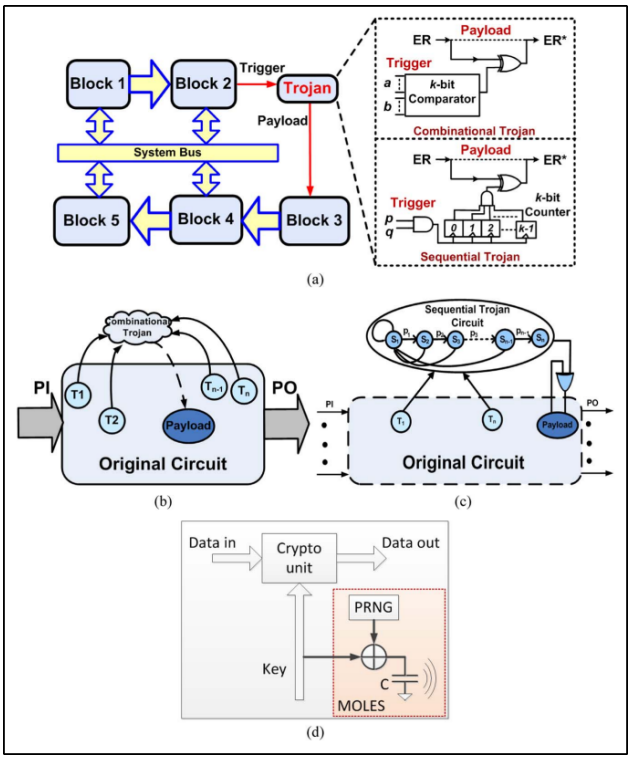

Figure 3.5: examples of different types of hardware Trojans

flipping signal ER). The sequential Trojan shown in Figure 3.5(a), on the other hand, undergoes a sequence of state transitions before triggering a malfunction. Figure 3.5(a) shows a synchronous k-bit counter, which activates when the count reaches 2k - 1, by modifying the node ER to an incorrect value at node ER*. Figure 3.5(b) and (c) show general models of combinational and sequential Trojans, respectively. These abstract models of Trojans are useful for studying the space of possible Trojans, and, similar to fault models, help in test vector generation for Trojan detection.

1) Trojans in Cryptographic Engines: A possible Trojan attack in a crypto engine can try to subvert the security mechanisms. The payload could range from a mechanism that presents dummy keys, predefined by the attacker, instead of the actual cryptographic keys used for sensitive encryption or signature verification operations, to leaking the secret hardware keys via covert side channels, e.g., information leaked through a power trace. Figure 3.5(d) provides an example of such a Trojan that attempts to leak a secret key from inside a cryptographic IC through power side-channels using a technique called malicious offchip leakage enabled by side channels (MOLES). Even if the IC has been designed to minimize side-channel information leakage, a hardware modification could help overcome the protection under specific circumstances where the attacker is in possession of the system or physically near the system to extract the secret information. Other targets could be a random number generator used for deriving random session

keys for a particular operation or the debug passwords used for unlocking test mode access to security-sensitive signals. Researchers have also proposed leaking such secret information over wireless channels by using low-bandwidth modulation of the transmitted signal.

2) Trojans in General-Purpose Processors: In general-purpose processors, an attacker at the fabrication facility can implement a backdoor, which can be exploited in the field by a software adversary. For example, modern processors implement a hardware chain of trust to ensure that malware cannot compromise the hardware assets such as secret keys and memory range protections. By using different stages of firmware and boot code authentication, one can ensure that the operating system (OS) kernel and lower levels (such as hypervisor) are not corrupted. However, in such systems, the attacker at an untrusted fabrication facility could implement a backdoor which disables the secure booting mechanism under certain rare conditions or when presented with a unique rare input condition in the hands of an end-user adversary. Similarly, other objectives which could be realized with the help of hardware Trojans would be to bypass memory range protections using buffer overflow attacks or to gain access to privileged assets by evading the access control protection mechanisms implemented in the hardware.

3.6.2 Trojan Taxonomy

The taxonomy of Trojan circuits has been presented in various forms, and it continues to evolve as newer attacks and Trojan types are discovered. Here, we will present a high-level classification, as shown in Fig. 4, based on variations in activation mechanism and Trojan effect. Based on the trigger condition, the hardware Trojans can be classified into analog and digital Trojans. The former are activated by analog conditions such as temperature, delay, or device aging effect, whereas the latter are triggered by some Boolean logic function. Digitally triggered Trojans can again be classified into combinational and sequential types. Analog Trojans include attacks on process steps that compromise reliability of all or select chips. These Trojans, referred to as reliability trojans by some researchers, may cause accelerated aging of the devices. The reduction in reliability is caused by acceleration of the wearing out mechanisms for complementary metal–oxide–semiconductor (CMOS) transistors, such as negative bias temperature instability (NBTI) or hot carrier injection (HCI). Selective malicious changes in the manufacturing process, such as variation in nitrate concentration in the gate oxide layer or the temperature used during the nitrate layering process can result in creation of infected ICs with a much shorter lifetime. What makes it challenging to detect such tampering is that it can be local to a particular functional circuit block or even to a small section of a block that can potentially evade standard postfabrication reliability characterization steps.

In terms of the payload, the Trojan can cause functional failure upon triggering or have a passive effect such as heating of the die or leaking of information. A Trojan can cause an “information leakage” attack, where secret information is leaked by a Trojan via a transmitted radio signal or serial data port interface such as the RS-232-C port. It could also involve a side-channel attack where the information is leaked through the power trace or through thermal radiation or through

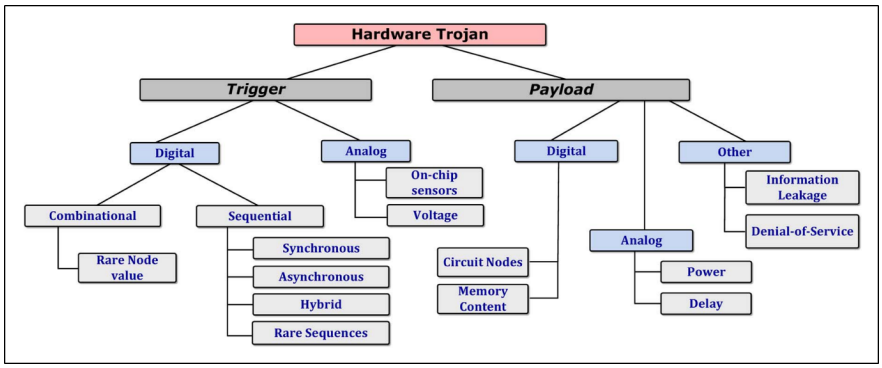

Figure 3.6: Trojan taxonomy based on trigger and payload mechanisms

optical modulation of an output light-emitting diode (LED). Another type of Trojan payload would be unauthorized alteration in system behavior, e.g., a denial-of-service (DoS) attack, which causes a system's functionality to be unavailable.

Other parameters used to define and compare different Trojans include the hardware overhead and activation probability. The area and power overhead relative to the design in which the Trojan is inserted has to be an undetectable fraction, in order to evade detection by obvious means. One can reuse existing logic and unused states in existing finite state machines (FSMs) to reduce the overhead. Moreover, the layout of complex ICs typically contains unused space which can be used for inserting extra gates without affecting the die footprint.

The activation probability is also a tradeoff between avoiding detection and malicious impact.

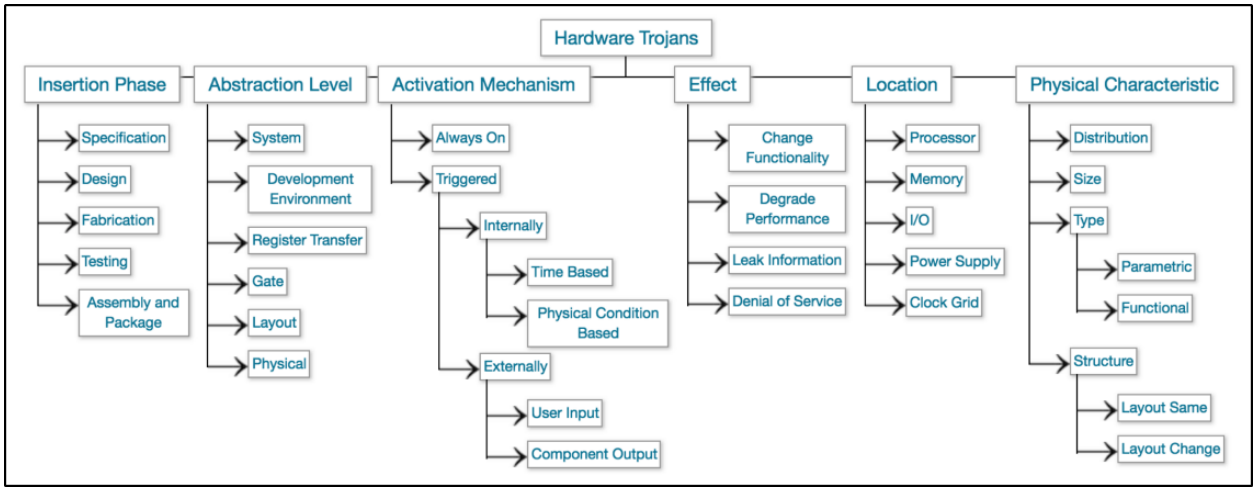

Figure 3.7: Trojan Taxonomy

Trojan Taxonomy Insertion Phase:

A Trojan can be introduced by changing the design specification, like temperature, to degrade the design dependability. Design and fabrication stages are also subjected to tampering. A Trojan can be realized by adding some extra gates to a design's netlist or by changing it's masks. Trojan insertion at the testing phase refers to trustworthy testing of a design after fabrication where an adversary may manipulate testing to keep an inserted Trojan undetected. Finally, unprotected interconnections between chips are prone to Trojan interference even if the chips are trustworthy by themselves. An unshielded wire connection could introduce unintended electromagnetic which an adversary can exploit for information leakage or fault injection.

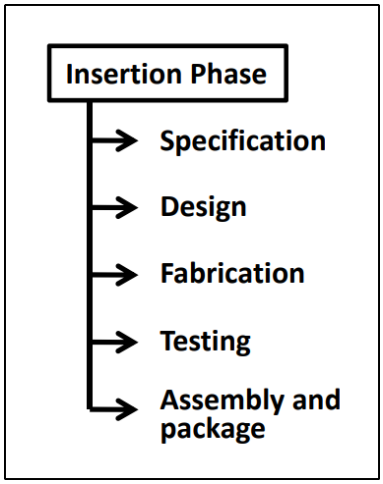

Figure 3.8: Insertion Phase

Trojan Taxonomy Abstraction Level:

The level of abstraction determines the control of advisory on Trojan Implementation. At the system level, a design is defined in terms of modules and interconnections between them, with an adversary being limited to the modules interfaces and their interactions. At the development environment level, a Trojan can be inserted into the modules by taking advantage of CAD tools and scripting languages. In more details, each module is described in terms of signals, registers and Boolean functions at the register-transrer level. In this level, an adversary has full access to the functionality and implementation of the modules and can easily change them. A design is represented as a list of gates and their interconnections at the gate level. Here, an adversary can implement Trojans in details, and Trojans' gates and their interconnections can be decided. At the layout level, the impact of Trojans on design power consumption or its delay characteristics can be controlled. Trojans can be realized even by changing the parameters of an original circuit's transistors. Finally, all circuit components and their dimensions and locations are determined in physical level. A Trojan can be inserted in white/dead space of the design layout with the least impact on the design characteristics.

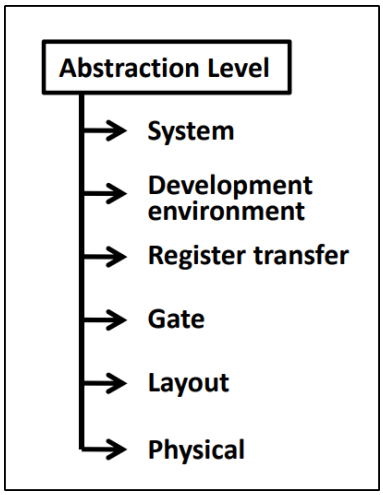

Figure 3.9: Abstraction level

Trojan Taxonomy Activation Mechanism:

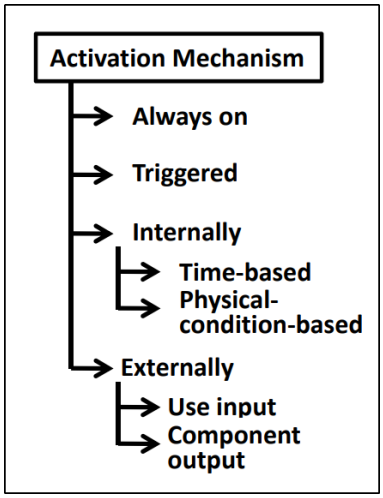

Figure 3.10: Activation mechanism

Trojans may always function, or they get conditionally activated. Always-on Trojans start as soon as their hosting designs are powered-on while conditional Trojans seek specific triggers either internally or externally to launch. The internal triggers can be timing-based (a Trojan is activated after certain time), or physical-condition-based (a Trojan is activated by certain events, e.g. specific temperature). The externally-triggered Trojans track user inputs or components' outputs, and the Trojans get activated if activation condition(s) are met.

Trojan Taxonomy Effect:

Trojans can be characterized based on their effects. The may change a design's functionality, for example, by modifying the data pa of processor. Trojans can reduce the design performance or degrade it reliability by changing the design parameters. A Trojan may leak the secret key of a cryptographic processor or can cause the denial of a service for an authorized requested service at specific time.

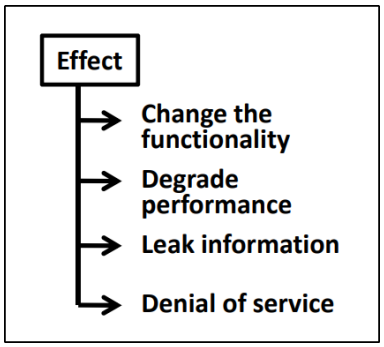

Figure 3.11: Effect

Trojan Taxonomy Location:

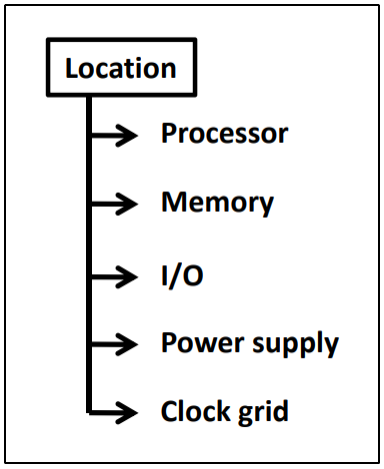

Figure 3.12: Location

Any part of a design is potentially subjected to Trojai. Inser A Trojan can be distributed over several parts or tightened in one part. A Tr can tamper with a processor to obtain control over its controller or data path units. A Trojan in a memory can change stored values or block read/woke accesses to the memory. On a Printed Circuit Board (PCB) including of several chips, an inserted Trojan on chips' interfaces can disturb communication. A Trojan can even affect the design power supply and alter current and voltage characteristics. Design delay characteristics can

change with interrupting the clock grid by a Trojan. The Trojan can freeze part of clock tree and disable some functional modules.

Trojan Taxonomy Physical Characteristics:

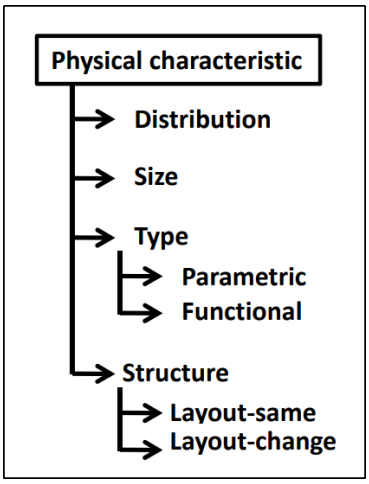

Figure 3.13: Physical characteristics

Trojan physical characteristics represent various hardware m estations. A Trojan can be a functional or parametric type where functio Trojans are realized by transistors/gates addition or deletion and parametric Trojans by wire thickness or any other design parameter modification. The number of transistors/gates added removed determines Trojan size. Trojan or distribution indicates how loose or tight Trojan cells are placed in the physical layout. Trojan structure refers to possible modification of original physical design for Trojan cells placements.

3.6.3 Taxonomy of board-level hardware trojans

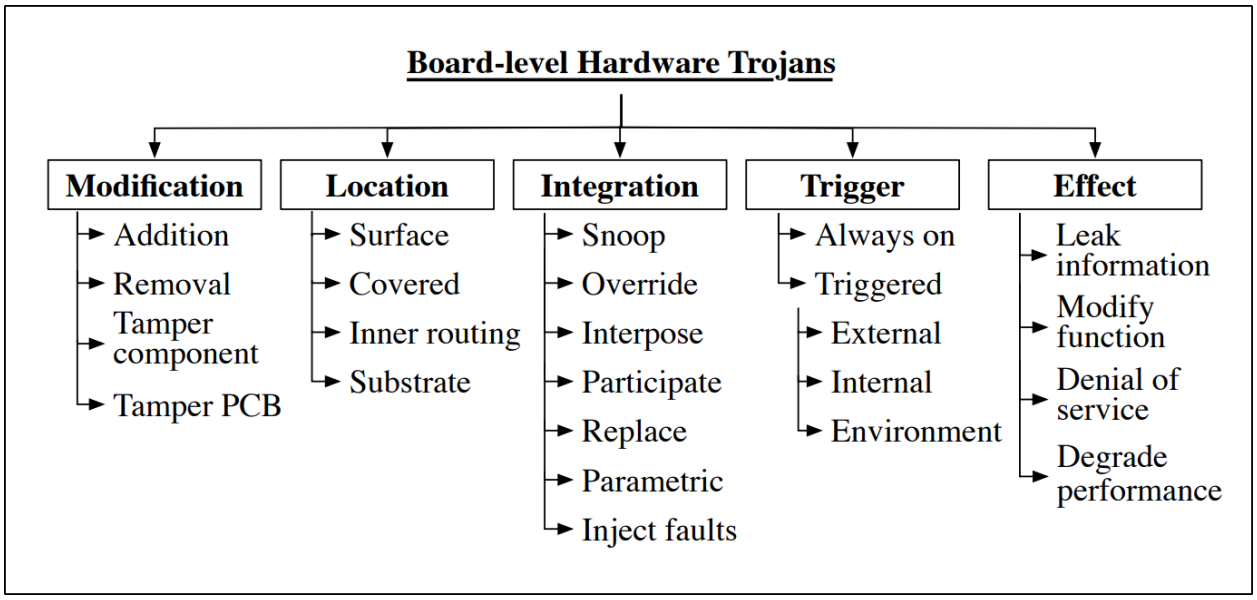

Figure 3.14: Taxonomy of board-level hardware trojans

Taxonomy of system-level trojans: Modification

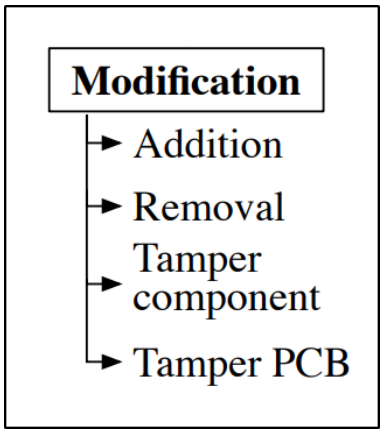

Figure 3.15: Modification

A Trojan's modification describes how a Trojan is made manifest in an electronic system.

A Trojan is realized through addition if an adversary adds components or traces that were not present in the legitimate system.

A Trojan may be realized by removing components to disable or compromise system functionality.

A tampered component describes a modification or replacement of a legitimate component.

Tampered PCB covers changes to routing or substrate.

Examples of modification:

Addition: Bloomberg's "The Big Hack" describes a Trojan IC that modified firmware as it was loaded by a SuperMicro server BMC. The chip was allegedly added to an unpopulated IC footprint.

Removal: One of several capacitors at the output of a switch mode power supply could be removed to destabilize power under heavy load.

Tamper component: Complex active components could be disguised as passive components.

Tamper PCB: Explores how copper traces could be thinned slightly to induce electromigration. explores how the location and dimensions of a PCB trace could be tampered to induce crosstalk with other traces.

Taxonomy of system-level trojans: Location

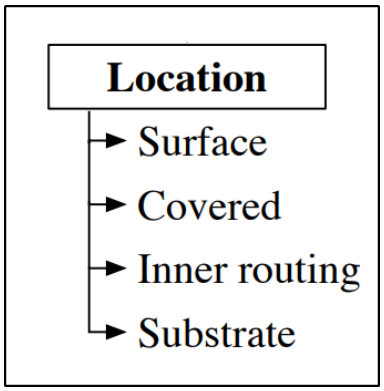

Figure 3.16: Taxonomy of system-level trojans: Location

The location of a PCB trojan influences how it might be detected and gives information about the sophistication of an attacker.

Surface trojans are located on the top or bottom of a system's PCB and are visible without moving other components.

Covered trojans are those that are hidden by a permanent fixture of a system such as a board marking, sticker, heatsink, or another component on a PCB. Covered components may also be hidden within component enclosures, e.g. within the metal casing of an ethernet socket.

Inter-routing trojans are hidden between the layers of a system's PCB. This includes embedded components and modifications to routing.

Substrate trojans modify the material composition of board substrate to influence dielectric or thermal properties.

Examples of location:

Surface: Demonstrates how a commercial single-board computer soldered onto the surface of a PCB can tap into various peripherals in a laptop dock to intercept and inject signals.

Covered: HOWLERMONKEY, as it is installed inside FIREWALK from the NSA ANT catalog [6], is an RF transceiver installed inside an ethernet socket.

Inner routing: shows how crosstalk can be exacerbated by slightly moving PCB traces: the magnitude of a coupled signal was increased enough to alter the logic value of a victim trace.

Substrate: Similar to inner routing Trojans, modifications to PCB substrate could degrade signal propagation or exacerbate crosstalk.

Taxonomy of system-level trojans: Integration

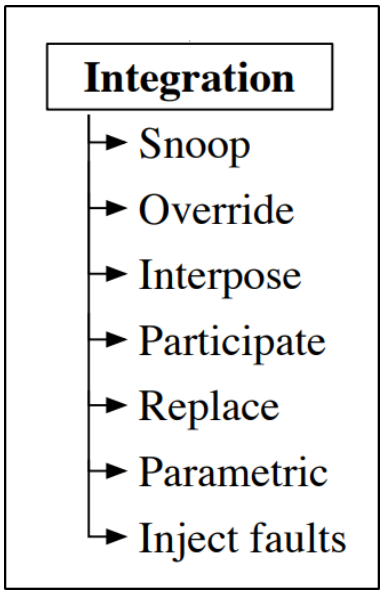

Figure 3.17: Integration

Integration describes how a trojan interacts with other components in an infected system and indicates what weakness in the system enabled the Trojan.

Trojans that **snoop** signals are those that read, but do not change, values of signals.

Trojans that **override** signals change them to influence a system's behavior.

A Trojan may **interpose** by breaking one signal from the legitimate system into two so that the signal can be controlled without creating contention

Replace covers scenarios where legitimate components are replaced by a Trojan component.

A Trojan can **participate** by "politely" interacting with other components in a system using legitimate protocols.

The **parametric** classification describes Trojans that influence passive characteristics, electromagnetic properties, trace impedance, heat dissipation, or other parameters.

A Trojan could force other components to operate outside their specifications and realize a payload by **injecting faults**.

Examples of integration:

Snoop: The NSA's SURLYSPAWN implant broadcasts the value of a snooped signal to remote adversaries via a radar retroreflector

Override: Implants proposed in read and override signals on a target's LPC or 12C bus to sabotage the security of TPM chips.

Interpose: A proof-of-concept for Bloomberg's "Big Hack" interposes between a flash chip and BMC to alter bits of firmware in transit.

Participate: PicoDMA is a PCIe exploit device connected to a wireless radio. It can be remotely controlled by an attacker to initiate DMA requests to read and write arbitrary memory on unprotected hosts.

Replace: demonstrates that ROM chips socketed onto a voting machine could be swapped quickly with chips containing election-rigging firmware.

Parametric: Adversaries could thin PCB traces to increase their resistance and induce premature electromigration.

Inject faults: Though we are aware of no board-level Trojans that have achieved this, Chip.fail demonstrates how secure boot protection in several microcontrollers can be bypassed by inducing voltage faults. Repackaging a device like Chip.fail into a standalone fault- injecting Trojan is an interesting research problem.

Taxonomy of system-level trojans: trigger

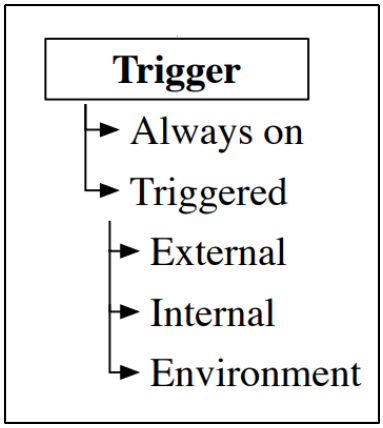

Figure 3.18: Trigger

A trojan's activation describes the conditions under which it delivers its payload.

Always active trojans have no "trigger"; they constantly deliver their payload whenever the infected system is active.

Triggered trojans deliver their payload when some condition is met.

• **Internal:** An internal trigger activates when an electronic system transmits certain signals or enters a certain state.

• **External:** An external trigger is a signal from an attacker that puts the system into a state that would otherwise have been unlikely.

• **Environmental:** An environmental trigger activates as the result of conditions unrelated to the primary function of a circuit. "Time bomb" triggers and presence of radiation are examples.

Examples of triggers:

Triggered:

• **Environmental:** PCB traces could be thinned to induce early system failure via electromigration

• **Internal:** discusses a Trojan chip that attaches to the serial port on a firewall: the Trojan chip emulates an administrator's computer to alter the firewall's configuration.

• **External:** FIREWALK is a Trojan ethernet socket that contains an RF transceiver to allow remotely triggered Trojan functionality.

Always Active: Replacing one passive component with another would result in an always active Trojan, as would a malicious modification to routing that causes a PCB trace to broadcast the value of a sensitive signal via electromagnetic radiation.

Taxonomy of system-level trojans: Effect

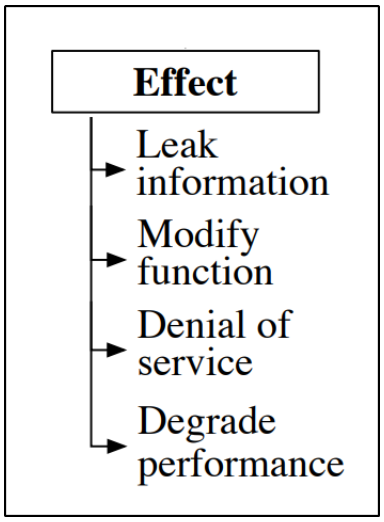

Figure 3.19: Effect

PCB Trojan payloads are classified in a manner that is consistent with IC Trojan taxonomies.

A Trojan may violate confidentiality by **leaking information**.

A Trojan that violates the integrity of a system **modifies function**.

Trojan attacks on availability are classified as **denial of service** payloads.

A Trojan may also **degrade performance** of a victim. An implant could alter the speed, power dissipation, or other performance characteristics of a circuit.

Examples of effect:

Leak information: SURLYSPAWN from the NSA ANT catalog taps low-frequency digital data (e.g. keystrokes from a keyboard) and broadcasts this data by modulating the switching frequency of a square wave in a manner that can be detected at distance by a radar ”retroreflector”.

Modify function: describes PCBs found in gas pumps in Brazil that were tampered to defraud customers by reporting dispensing more fuel than they had actually dispensed.

Denial of service: demonstrates how removing a resistor from an industrial fan controller causes

a decoupling capacitor to slowly charge and eventually disable control of the fan. Trojans could also permanently damage PCBs by cauing over-current, over-voltage, over-heating, or other extreme circuit conditions.

Degrade performance: The addition of a resistor that continuously dissipates power in order to drain a battery could be said to degrade system performance.

3.7 Issues with Third party IP Design:

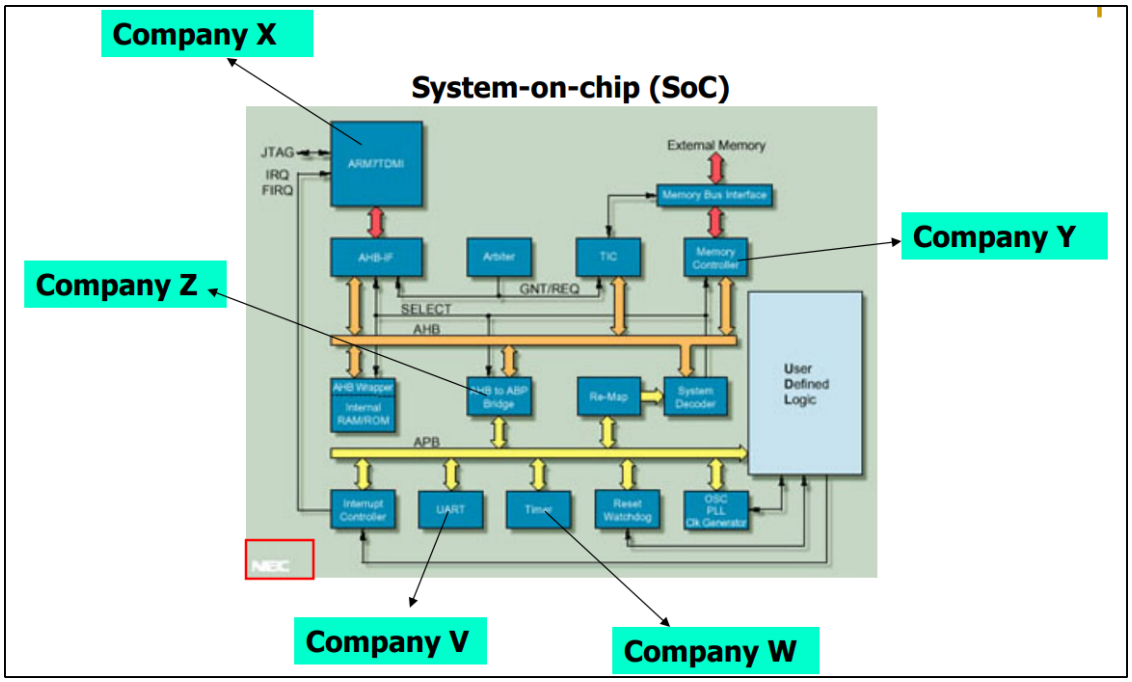

Figure 3.20: Issues with third party IP design

The use of third-party intellectual property (IP) is becoming increasingly common in the semiconductor industry as it can reduce design costs and accelerate product development. However, the use of third-party IP comes with its own set of issues, particularly related to hardware Trojan detection.

One of the most significant security concerns related to using different parts made by different companies is the possibility of a hardware Trojan. A hardware Trojan is an intentional alteration made to a design that can cause it to behave maliciously. It can be inserted at any point in the supply chain, from the design phase to the manufacturing and testing phases. Hardware Trojans are designed to remain hidden and activate under specific conditions, making them difficult to detect. They can be inserted into the design of a semiconductor component by a third-party vendor, who may have malicious intent or may have been compromised by an attacker. In

addition, different parts made by different companies can have different levels of security, making it difficult to ensure that the entire system is secure.

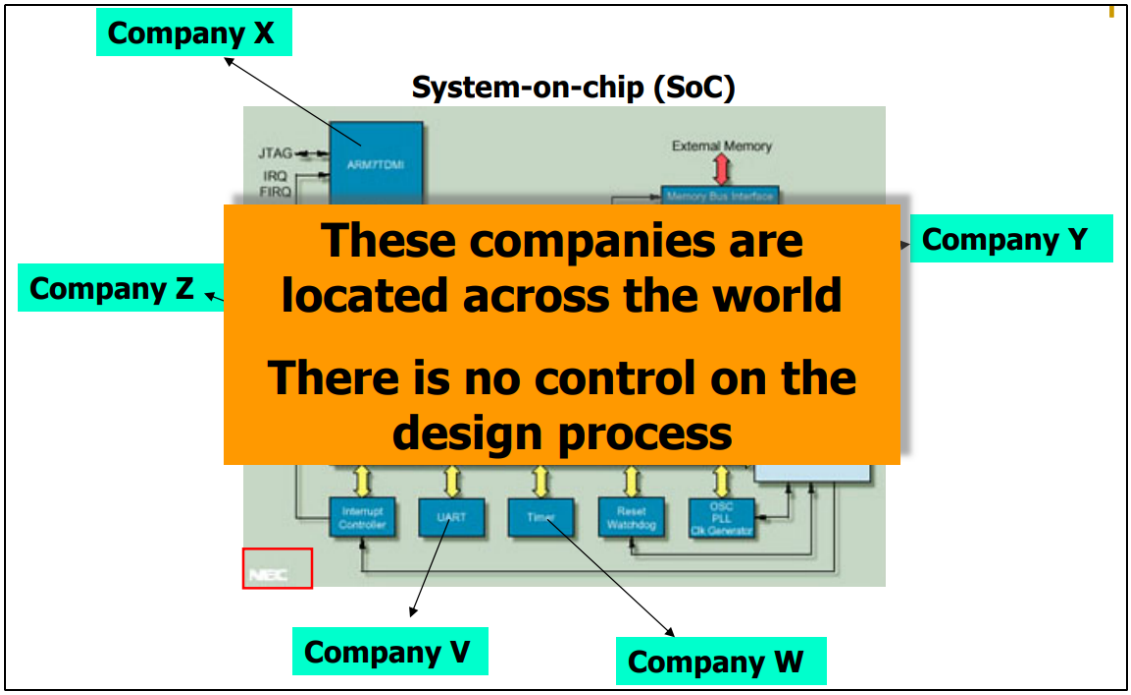

Figure 3.21: Issues with third party IP design

Counterfeit parts are also a significant security concern related to using different parts made by different companies. Counterfeit parts can be used to insert hardware Trojans or other malicious code into a system, or they may simply not function correctly, leading to system failures or data breaches.

Supply chain attacks are another security concern related to using different parts made by different companies. In a supply chain attack, an attacker targets a company in the supply chain to gain access to the entire system. Supply chain attacks can be difficult to detect and can be devastating, as they can compromise the security of the entire system.

The lack of transparency from third-party IP vendors is another issue related to hardware Trojan detection. Third-party IP vendors often do not provide detailed information about their IP designs, making it difficult for designers to understand the internal workings of the IP and detect any potential hardware Trojan.

Integrating third-party IP into a larger design can also be challenging. The third-party IP may not be compatible with the rest of the design, leading to integration issues that can be difficult to detect and troubleshoot. Furthermore, the use of third-party IP creates a trust issue, as designers

must rely on the IP vendor to ensure that the IP does not contain any malicious hardware Trojan. This lack of trust can create significant security concerns for designers.

Companies in the semiconductor industry must take a proactive approach to security to address these concerns. This includes conducting thorough risk assessments of all third-party vendors and suppliers, verifying the authenticity of all components used in their products, and implementing robust security measures throughout the supply chain. A security framework that includes security requirements for all components used in a product should be implemented to ensure that all components meet the necessary security standards.

In conclusion, the use of third-party IP in the semiconductor industry has its own set of security concerns, including the insertion of hardware Trojans, counterfeit parts, and supply chain attacks. Addressing these concerns requires a proactive approach to security, including thorough risk assessments, implementing robust security measures throughout the supply chain, and working closely with third-party vendors and suppliers to ensure that they are following best practices for security.

3.8 Instances of Hardware Trojans in the Wild

In recent years, there have been several instances of hardware Trojans being discovered in the wild. Some of these instances are discussed below.

Stuxnet Worm: Stuxnet was a highly sophisticated computer worm that was discovered in 2010. It is believed to have been developed jointly by the US and Israel and was designed to target Iran's nuclear program. The worm targeted a specific Siemens industrial control system and modified the code to cause damage to centrifuges used in uranium enrichment. Stuxnet is considered to be one of the most advanced pieces of malware ever discovered, and its discovery raised concerns about the potential for cyber attacks on critical infrastructure.

Infineon TPM Vulnerability: In 2017, a vulnerability was discovered in the Trusted Platform Module (TPM) used by Infineon. The vulnerability allowed an attacker to generate a master RSA key for any TPM chip manufactured by Infineon. This key could then be used to decrypt data encrypted with the corresponding public key. This vulnerability was the result of a hardware Trojan that had been introduced into the Infineon TPM during the manufacturing process.

Lenovo Superfish: In 2015, it was discovered that Lenovo had pre-installed a piece of software called Superfish on its laptops. Superfish was designed to inject ads into web pages, but it also introduced a security vulnerability by installing a self-signed root certificate on the system. This certificate could be used to intercept encrypted traffic and steal sensitive information. It was later revealed that Superfish had been designed to bypass HTTPS encryption and was a result of a hardware Trojan that had been introduced during the manufacturing process.

Cisco Router Backdoor: In 2018, a backdoor was discovered in some Cisco routers. The backdoor allowed an attacker to remotely access the router without authentication and execute arbitrary commands on the device. This backdoor was the result of a hardware Trojan that had been introduced during the manufacturing process.

Apple iPhone BatteryGate: In 2017, Apple was accused of intentionally slowing down older iPhone models to encourage users to upgrade to newer models. It was later discovered that the slowdown was due to a software update that was designed to prevent the phones from shutting down unexpectedly due to aging batteries. However, some users believed that this was a result of a hardware Trojan that had been introduced into the iPhone batteries.

Starbug: Starbug is a hacker who has gained notoriety for his ability to hack into biometric systems. In 2014, he demonstrated that it was possible to bypass the fingerprint scanner on the iPhone 5S by using a fake fingerprint. This was achieved by creating a mold of a fingerprint using wood glue and then using the mold to create a fake fingerprint that could be used to unlock the phone. This attack was a result of a hardware Trojan that had been introduced into the iPhone's fingerprint scanner.

Syrian Electronic Army: The Syrian Electronic Army (SEA) is a group of hackers who support the Syrian government. In 2013, the SEA hacked into the Twitter account of the Associated Press (AP) and posted a fake tweet claiming that there had been an explosion at the White House and that President Obama had been injured. This caused a brief panic in the financial markets, and the Dow Jones Industrial Average dropped by over 150 points. The attack was a result of a hardware Trojan that had been introduced into the AP's Twitter account.

Chapter 4

PHYSICAL UNCLONABLE FUNCTIONS (PUFs)

In this Modern security system importance of authentication and safety of information been drastically increased with the extensive usage of personal devices with secret keys. The conventional security has created protocols and algorithms which are stored in the non-volatile memory for cryptographic applications which is very slow and consumes more power. In addition to that, the non-volatile memories have untrusted environment and attacker has high chances to extract physical information of the system.

Over the last few years, the authentication through physical unclonable functions (PUFs) has risen significantly, making them a hot topic in the hardware security domain. The prediction or extraction of secret key is extremely challenging in PUF circuits because of random variations during the fabrication of ICs which are oxide thickness, threshold voltage and capacitances. This method is beneficial than conventional digital security approach. PUF uses simple circuits which are easy to manufacture, consumes less area, and dissipates minimum power than conventional security circuits. However, the fabrication of nonvolatile memories are very affluent and it requires continual power supply. In addition to that, it requires complex algorithms such as secure hash or public/private key encryption algorithm and expensive cryptographic hardware. Whereas the PUF circuits need not require complex algorithms and hardware structure.

4.1 Characteristics and Parameters of PUF Circuits

The characteristics of PUF circuits are based on challenge-response pairs, inter and intra distance measures and environmental effects are described as follows.

A) Challenges and Responses The PUF can be exhibited as a challenge-response system which is shown in Fig.1 as an input challenge ‘C’ is passed to a PUF and an output response ‘R’ where

R = f(C). It is difficult to replicate similar challenge-responses pairs (CRP) with two different PUF circuits and only an authentic IC can produce a correct response for a challenge.

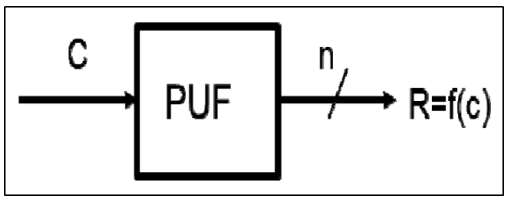

Figure 4.1: Basic representation of PUF system

B) Inter and Intra Distance Measure The inter-distance for a particular challenge 'C' between two different PUF circuits 'PUF1' and 'PUF2' are the distance between two responses 'R1' and 'R2' is shown in Fig. 2. Inter-distance should be as close to 50% as possible for reliable PUF.

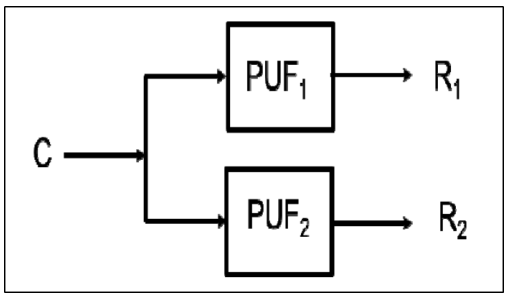

Figure 4.2: Symbolic representation of inter-distance between two PUFs

The intra-distance between two evaluations on one single PUF is the distance between the two responses 'R1' and 'R2' resulting from a particular challenge 'C' which is depicted in the Fig.3. Intra distances should be as small as possible for reliable PUF.

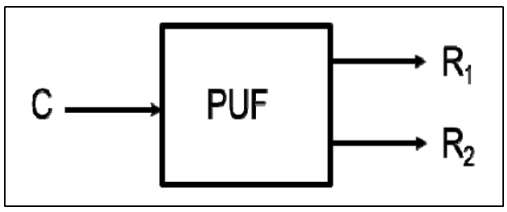

Figure 4.3: Symbolic representation of intra-distance of a single PUF

C) Environmental Effects A PUF responses generally involves a physical measurement, there are a number of unwanted physical side effects could interface such as the random noise and measurement uncertainties. The temperature and supply voltage variations affect the measurement of PUF responses on an IC. The basic parameters of the PUF circuits are based on the uniqueness, reliability, uniformity and bit-aliasing.

D) Uniqueness Uniqueness measures the ability of PUF circuits among a group of ICs which is similar. To find the uniqueness between two different PUFs, the hamming distance (HD) technique can be implemented and it also measures the inter-distance responses between the pair of PUF circuits. If the two PUF circuits 'i' and 'j' have responses 'Ri' and 'Rj' respectively for a challenge 'C', the average inter-chip HD among 'k' circuits is as follows.

$$Uniqueness = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{HD(R_i R_j)}{n} \times 100\%$$

E) Reliability The reliability of PUF circuits are used to measure the reproducing efficiency of a PUF with n-response bits at various operating conditions like temperature and supply voltage with 'm' samples of R'i, value. For a PUF circuit 'i', the intra-chip HD is estimated as follows.

$$HD_{INTRA} = \frac{1}{m} \sum_{i=1}^{m} \frac{HD(R_i, R'_{i,t})}{n} \times 100\%$$

where $R'_{i,t}$ is the t^{th} sample of R'_i and HD_{INTRA} indicates the average number of n-response bits.

The actual reliability of a PUF circuit is defined as

$$Reliability = 100\% - HD_{INTRA}$$

F) Uniformity Uniformity is a measure of the unpredictability or randomness of a PUF's response. A uniform distribution of 1's and 0's ensures a strong security key which is difficult to replicate. The actual PUF responses is proportional to 50%. The uniformity of n-bit PUF response.

H) Bit-aliasing The bit-aliasing is used to measure the bias of a particular response bit across different PUF circuits. The Hamming-Weight of the l-th bit of the identifier across k-devices.

4.2 Types of PUF Models and Applications

Based on the number of challenges-response pairs (CRPs) strong and the weak PUFs are categorized which are used for low-cost authentication and to store secret keys.

4.2.1 Weak PUF Model

The weak PUF can only be supported by one or a small number of challenges. It's response is stable and robust to environmental conditions. However, It has only a limited number of CRPs,

these pairs must be kept secret because its output response can be used as a secret key to encrypt/decrypt data of the device. The weak PUF requires additional cryptographic hardware to provide authentication services.

Applications of Weak PUF Model

i) Ring-Oscillator PUF

Ring oscillator contains N identically designed oscillators which has N(N-1)/2 possible pairings. It measures differences in gate delay and it is also susceptible to the environmental variations. The 'close' frequency oscillators causes more error than the frequencies that are far apart. It uses index-based error correcting technique to improve the stability. The representation of the ring oscillator PUF.

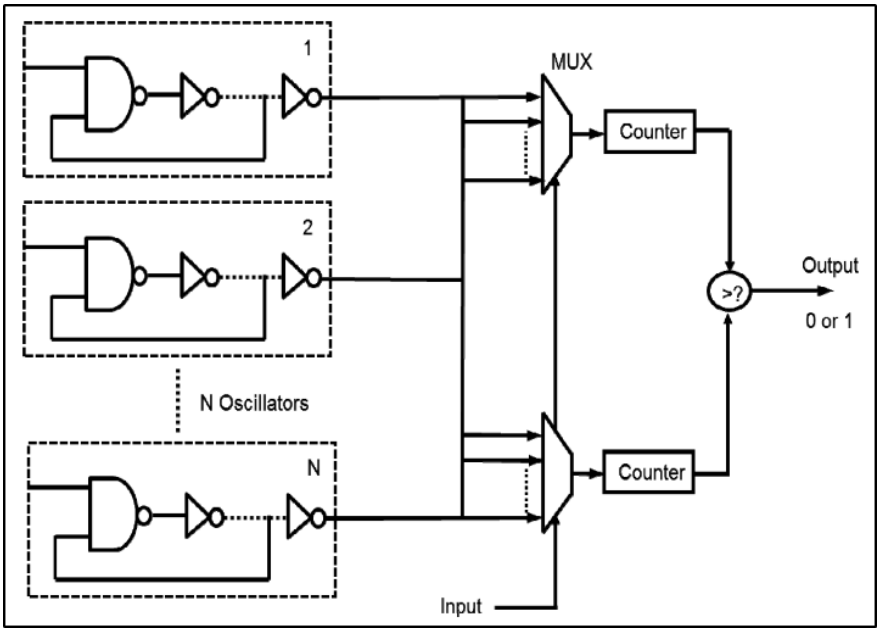

Figure 4.4: Schematic of Ring Oscillator PUF

ii) SRAM PUF

The Figure 4.5 shows the schematic representation of SRAM PUF and it is composed of two cross coupled inverters with the stable states of '1' and '0'. If device powers-up, the feedback pushing the cell toward the "1" state equals the feedback pushing the cell toward the "0" state, thereby keeping the cell in this metastable state indefinitely.

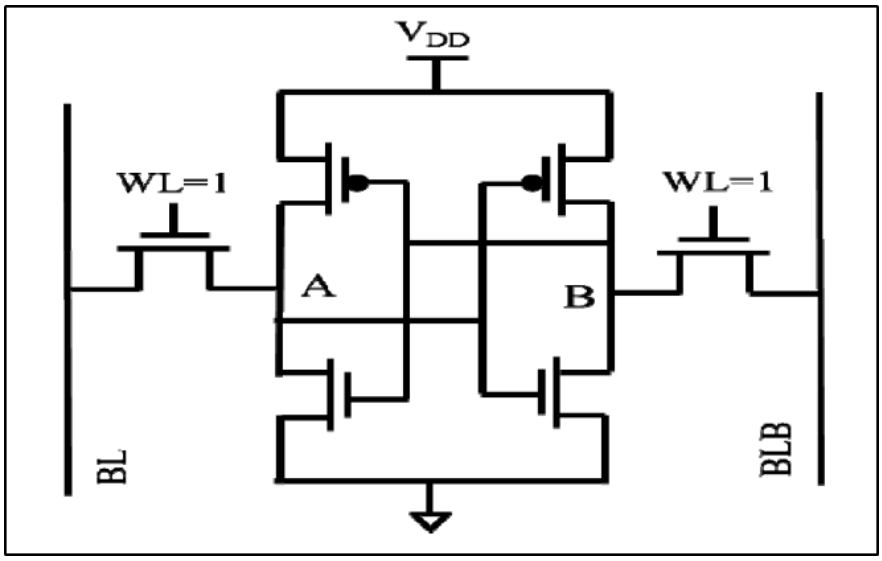

Figure 4.5: Schematic of SRAM PUF

4.2.2 Strong PUF Model

Strong PUFs can support a large number of challenge-response pairs such that an attacker cannot compute all CRPs within a fixed timeframe. The additional crypto hardware is not required for authentication and secret key generation. Hence the output responses of the strong PUFs are not necessary to kept secret.

Applications of Strong PUF Model

i) Optical PUF

In optical PUF, the input is a laser and the output response is associated with speckle patterns. These patterns are dependent on input which is composed of large number of randomly positioned 100um silica spheres. Each sphere act like a small lens and it refract the light rays as they move towards scattering block. The basic implementation of the optical PUF is represented in Figure 4.6

ii) Arbiter PUF

In Arbiter PUF, depending upon the challenge bits the input edge is split to two MUXs. Fig. 7 represents the basic operation and schematic implementation of an arbiter PUF. The output of each MUX will be given to the latch and each latch acts as an arbiter. Even with direct physical access to arbiter PUF the attacker find difficult to measure the individual gate delays. By measuring the variations like supply voltage, temperature and noise, adversary can affect the arbiter PUF. The error correcting techniques like 2D-hamming code can be implemented to

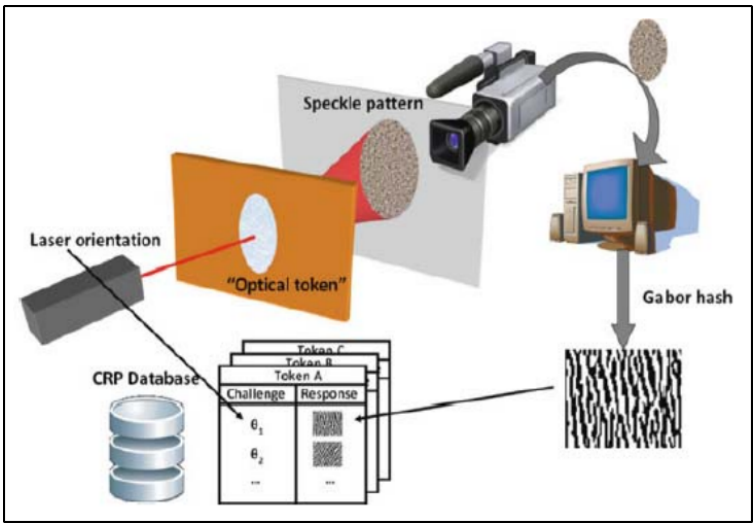

Figure 4.6: Basic implementation of an optical PUF

improve the stability and security of the arbiter PUF circuit.

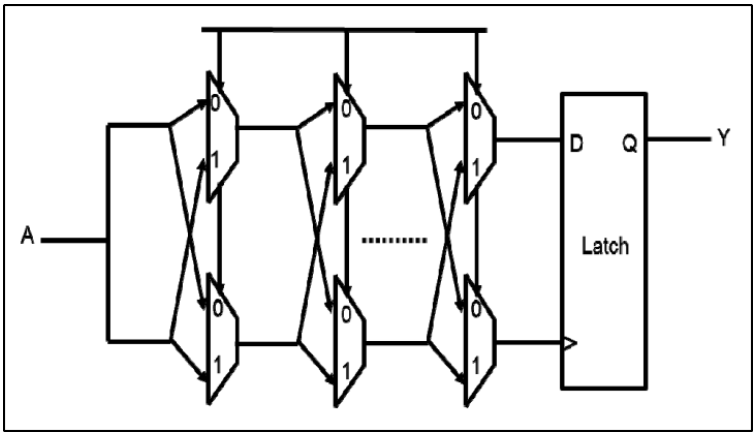

Figure 4.7: Schematic representation of arbiter PUF

4.2.3 Physical Unclonable Function Architecture Selection

Selecting an appropriate PUF architecture for any applications is not trivial. The main criteria for this selection are:

1. Robustness against possible attacks.

2. Good statistical properties are important features in cryptographic applications (CRPs uniqueness and uniformity).

PUF Architecture	Strength	Weakness	FPGA compatibility
Arbiter PUF	(Partly) Resilience against machine learning attack.	Delay path must be identical	No
Ring Oscillator PUF	Easy to implement	Environmental sensitivity	Yes
SRAM PUF	Good statistical properties	Low number of CRPs	No
Public PUF	No secret info	Need further investigations	Need further investigation
TERO PUF	Environmental resiliency	Need further investigations	Need further investigations
Other new PUF (Interpose PUF, LRR-DPUF, array PUF)	Resiliency against machine learning attack	Need further investigations	Need further investigations

Figure 4.8: PUF architecture summary table. CRP, Challenge Response Pair; TERO, Transient Effect Ring Oscillator; LRR-DPUF, learning resilient and reliable digital PUF.

3. Number of CRPs vs. occupied area (strong PUF: exponential increment in the number of CRPs by increasing utilized computational resources vs. weak PUF: linear increment in the number of CRPs by increasing utilized computational resources)

4. Easy implementation process on FPGA (making it possible to adapt the PUF after deploying the IoT devices and updating it to new counter attacks).

In this section, we discuss the proposed PUF architectures for IoT applications, and we look into the strengths and weaknesses of each architecture. Figure 4.8 provides an overview.

4.2.4 PUF Protocols

Different authentication protocols that use a PUF have been proposed. In the following, we give an overview of existing protocols, taking into account different features that they support. We mainly focus on protocols that have been proposed explicitly for the IoT, but also take into account protocols that provide a feature that we find important for the IoT and that may be applied to it. A short summary is given in Figure 9. Protocols are often not independent of the type of PUF (weak or strong) that they use. Protocols based on weak PUFs [16,40,70] typically require cryptographic methods (e.g., hashing, encryption, etc.) to compensate for CRP scarcity. This mostly negates the advantage of using a PUF in IoT. Therefore, we focus on protocols that use strong PUFs in this paper. The first authentication protocol that used a (strong) PUF [15] (as described in Section 2) was based on the original assumption of a perfect PUF that can never be cloned or modeled [70]. As further research showed (see Section 3) this assumption proved to be

PUF Protocols	Strength	Weakness	Compatible to which PUF?	FPGA Compatibility	Further Notice
Early protocols	(Partly) Resilience against machine learning attacks	Cryptographic primitives utilization	Integrated into different PUF architectures	No	Occupy considerable amount of computational resources
Mutual authentication protocol	Mutual authentication feature	Assumption of perfect PUF architecture	Not specified	Need further investigation	Implementation overhead has not been reported
Obfuscated challenge response protocol	Resiliency against machine learning attacks	Dependency on random number generation	Arbiter PUF	Need further investigation	Computational resource utilization & FPGA compatibility need to be clarified
Lockdown protocol	Resiliency against machine learning attacks	Need further investigation	XOR Arbiter PUF	Need further investigation	Computational resource utilization needs to be clarified

Figure 4.9: PUF protocols

incorrect. Especially when faced with machine learning analysis of CRPs [22], PUFs showed how fragile they can be. The most important task of a PUF authentication protocol therefore is to be robust against machine learning attacks.

Another important feature of PUF protocols is the ability to cope with (partially) unstable PUFs. Unlike initial PUF definitions suggest, responses are often not 100need error correction methods to cope with this [71]. Finally, it is especially important for IoT systems to support mutual authentication. Both communication partners should be authenticated against each other. Otherwise, IoT sensors might, e.g., send personal data about users to untrustworthy (unauthenticated) servers, and servers might accept faked sensor measurements from (unauthenticated) attackers. Since IoT devices often communicate directly, ideally, mutual authentication should be provided, not just between a device and a server, but also between two devices.

4.3 Emerging PUF with Nanoelectronics

The next generation PUF circuits are implemented using emerging nanoelectronic devices such as carbon-nanotube-field-effect transistors (CNTFETs) and memristors have several levels of inhibit randomness due to number of fabrication process variations like doping profile, thickness and area as device scales to the nanometer level makes these devices suitable to derive secret keys and authentication.

4.3.1 CNTFET Based PUF

The CNTFET is a promising nanoelectronic based device which overcomes some of the limitations in the conventional MOSFETs. The schematic representation of CNTFET based PUF is shown in Fig.8 has two rows of CNTFETs connected to the comparator and generate one response bit. The variations between every individual CNTFETs are unique. The CNTFET based PUF dissipates less power and limited area than the conventional PUFs.

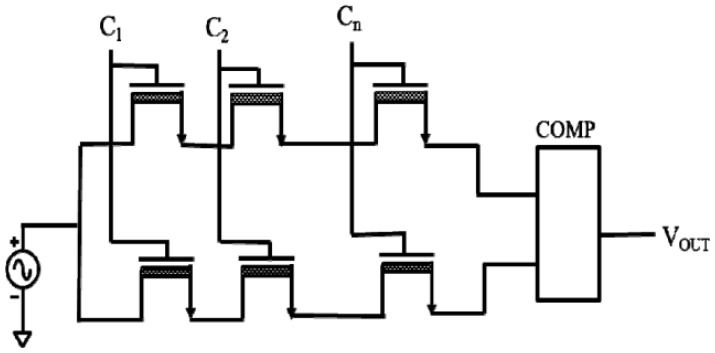

Figure 4.10: Schematic design of CNTFET based PUF

4.3.2 Memristor Based PUF

Memristive PUFs are designed by interconnecting memristors in a cross-bar pattern which is shown in Figure 4.10 and these PUFs uses write-time irregularities. The memristor based PUFs dissipates less power and are resilient to power analysis-based side- channel attacks.

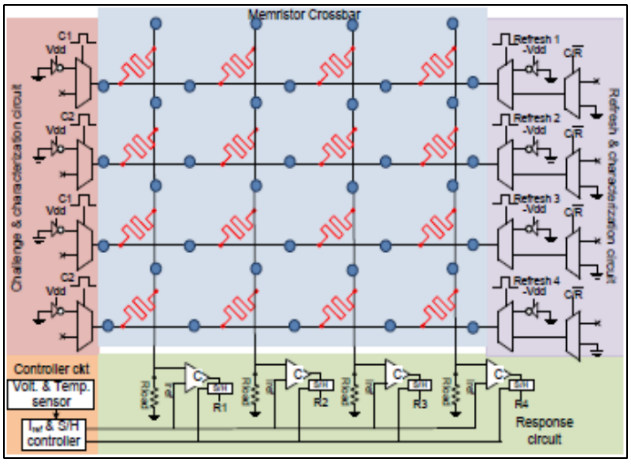

Figure 4.11: Memristive based PUF

4.4 PUF-Based Applications

Due to the unique properties of PUFs, they have been used in a large variety of applications such as integrated circuit intellectual property (IP) protection, key generation, device authentication, digital rights management, trusted computing, vehicle system security, and so on. In this section, we first briefly survey two of PUF's most popular and well-documented applications, namely secure key storage and device authentication. Then we give a detailed introduction on how PUF can help in IP protection. We must note that PUF responses may change due to factors such as ambient temperature variation and supply voltage fluctuation since these factors may affect circuit delay in practice, and hence reliability-enhancing techniques such as string pattern matching, Index-Based Syndrome coding (IBS) or fuzzy extractor need to be used for different applications.

4.4.1 Secure Key Storage

With the popularization of electronic devices, we are increasingly dependent on IC to securely handle sensitive information. For example, the RFID is used as a key card to control the building, and smart cards are used to perform financial transactions. Therefore, it is important that IC can protect sensitive information. The traditional method is to store a secure key in the non-volatile memory (e.g., EEPROM) in order to use cryptographic primitives (such as digital signatures and encryption) to protect sensitive information. However, it has some obvious drawbacks. For example, the recently proposed non-invasive and invasive physical tampering techniques (e.g., micro-probing, laser cutting, side-channel) can allow an attacker to extract the digital key stored in nonvolatile memory, and therefore compromise the cryptographic-related secure mechanisms. In order to prevent physical attacks, researchers have proposed various protection mechanisms, where PUF is a new cryptographic primitive that can produce a secure volatile key. By means of storing secrets in its unique intrinsic physical features that are randomly determined by fabrication variations, e.g., the subtle difference in the delays of two wires with equal length at the design phase, PUF achieves a higher level of protection without relying on persistent power. The validity of the claim rests on the insight that attempts in conducting invasive attacks will alter the unique intrinsic features and therefore destroy the secret hidden in the victim devices with a higher probability.

4.4.2 Device Authentication

Strong PUFs exhibit a huge number of challenge-response pairs (CRPs), hence they can be used for device authentication with low cost. Suh et al. [33] demonstrated how PUFs can be used to authenticate individual ICs without costly cryptographic primitives. As shown in Fig.5, a PUF is embedded into the device A, and CRPs are collected and stored in a secure database. As the PUF responses are unique and unpredictable for each device, we can simply compare a re-generated PUF response with the pre-stored response in the database with the same challenge. When there is a perfect match or the matched portion is more than a pre-determined threshold, the device will be considered authenticated. In order to protect against man-in-the-middle attacks, the used CRPs

will be deleted from the database. Later on, many lightweight and secure PUF authentication protocols are also proposed. These protocols are suited to resource-constrained environment such as embedded systems and pervasive networks.

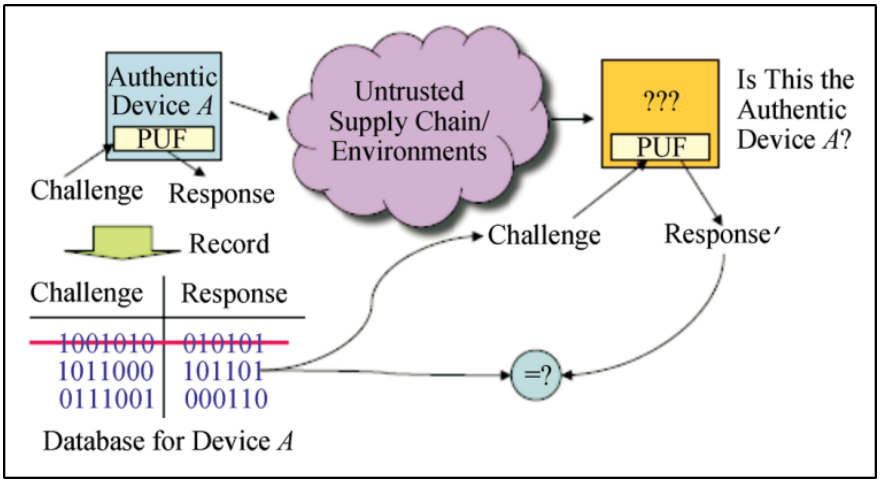

Figure 4.12: PUF-based low-cost authentication

4.4.3 FPGA IP Protection

With FPGA's static and dynamic reconfigurability, continuous improvement in performance, and the decrease of production cost, FPGAs have been widely used in the computing acceleration, communication, and other areas. However, hardware IP (HW-IP) which is defined as the soft-core (synthesized from HDL) hardware modules stored in the FPGA configuration bitstream is vulnerable to piracy attacks. PUFs have been used as a promising hardware security primitive to resolve the issue. Currently, several HW-IP protection techniques have been proposed to prevent piracy such as watermarking[80-82] and encryption. However, watermarking is a passive IP protection technology, which means that it cannot actively prevent IP from being illegally duplicated, distributed, or integrated into SOC, and encryption-based methods are also faced with some severe issues. Most recently, Zhang et al. proposed the first nonencryption-based FPGA IP binding technique that combines the unclonable PUF signatures of hardware with the finite state machines (FSMs) of sequential circuits in FPGA designs/IP cores to actively restrict FPGA designs/IPs to running on authorized hardware platforms. The schemes can potentially address the drawbacks of the encryption-based schemes mentioned above. Meanwhile, it can provide commercially popular pay-per-device licensing mechanism which provides technical support for the system developers to pay IP licensing fees only for the FPGA devices they are using. The key part of the binding method is the interaction protocol among the FPGA vendor, core vendor, system developer, and end user. The binding protocol includes four parts: 1) FPGA device enrollment; 2) HW-IP core enrollment and distributing; 3) HW-IP core licensing; 4) FPGA-based product licensing. Based on the binding protocol, the authors introduced a prototyping design and implementation of the lock mechanism proposed in the binding scheme.

Chapter 5

RECENT DETECTION METHODS OF THE HARDWARE TROJANS

Hardware security is a kind of vulnerability protection, which appears in the form of physical equipment, not in the form of software installed on computer system hardware. It may be related to the device used to scan the system (or) monitor network traffic. It is a physical device rather than a software form of vulnerability protection isolated on computer system hardware. The term hardware security also refers to protecting the physical system from damage. In the current internet world, security is a major issue in software and hardware implementations. The growing demand for consumer electronic products requires multiple design cycles. To speed up the design cycle, the company is seeking common application IP protocols with third parties, such as USB, encryption, DSP, etc. These third parties may introduce malicious content, namely trojan. trojan horses in the netlist can only be activated using special inputs/triggers. The available Trojan horse detection techniques Delay, area, power fingerprint technology, and automatic test pattern generator (ATPG) methods) are not recommended because of more detecting time with less accuracy.

Inserting a hardware Trojan horse can potentially allow an attacker to obtain valuable information, such as encryption keys, or prevent the correct operation of the design through a denial of service attack. During the testing and verification process, it is usually almost impossible to detect hardware Trojans because they are triggered by an attacker only under a very specific set of circumstances. It is not dependent on hardware provided, whether any electronic gadgets may be just a hardware and software integrated system, several threats behind it, in any supply chain there is some backdoors at various levels (multiple levels) of the system. The designer must also use a third party (intellectual Property Rights) IPR when designing the system. We are in need to develop an identification tool to ensure the trustiness of the HT-free design. The reason is that when people design hardware to implement applications and in various functions in primitives, they consider performance rather than security. Due to this poor security testing

analysis, there may be malicious designers and malicious foundries. They may add hardware Trojans to the hardware. In addition to these issues, related to the hardware designer's process, the hardware itself is subjected to attacks such as side-channel analysis and physical attacks.

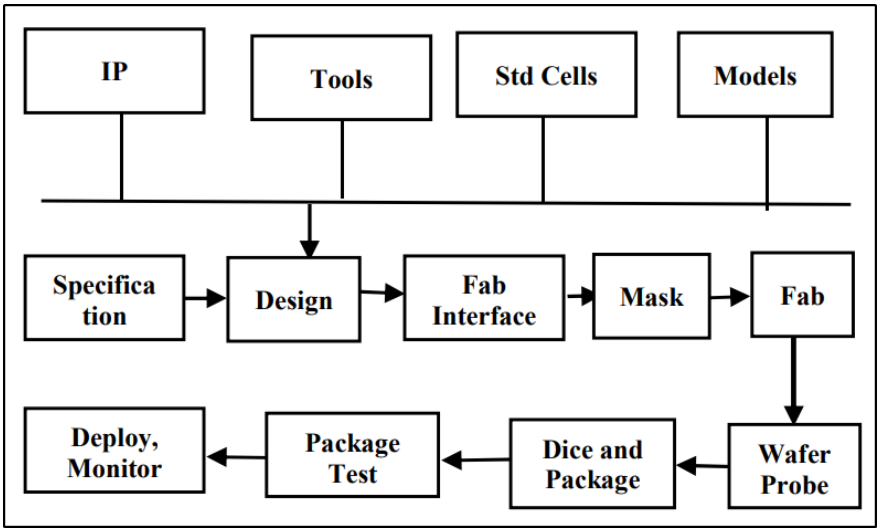

Figure 5.1: Vulnerable steps of ICs

Vulnerable steps of ICs are mainly classified into three parameters (i) Trusted, (ii) Either, (iii) Untrusted. The trusted (secured) parts in the IC are specification, deploy and monitor and package test. Either part mask and fabrication (FAB) areas. Untrusted (not secured) parts in the IC are IP, tools, std cells, models, design, fabrication interface, wafer probe, dice, and package. It is a malicious change (or) addition to integrated circuits it's added (or) removes performance (or) reduces flexibility.

Malicious and unwanted modifications done to gadgets like chips and integrated circuits. So it affects the performance of the chips. properties of hardware Trojan: a. very small, b. need few mux and trigger signals, c. mostly passive, d. it can be added in multiple stages, e. the main objective of hardware Trojan extremely very small so prediction is complex. Trojan can be inserted anywhere during the manufacturing process (e.g., in third-party IP cores purchased, by fabrication plant, etc.).

Hardware Trojan is generally injected by Manufacturing, Design manipulation third party, during shipment and Built-in levels (Back doors). Moreover, the major two primary injections are trigger circuit and payload. In trigger circuit Based on a selection occurring in the event (Internally activated, always-on, Condition-based, externally activated, Antenna and sensors). Ex: particularly rare packet arrives in the network. Payload is based on modify function, leak information, degrade performance. Hardware Trojan insertion modifies the circuit function, disclosure of confidential information, and destroys the chip. Mainly to develop reliable HT identification and defense models, primarily to understand the feasibility of injecting hardware

Trojans in the actual implementation. More specifically, how to insert the invisible HT into the target circuit, how feasible is the HT for a specific application model, and what are the challenges to achieving such HT. Some hardware Trojan were small and normal size, it may add at register transfer level.

5.1 Hardware Trojan Detection

5.1.1 Pre-Deployment Hardware Trojan Detection

Operating system malware, Trojan horses that cannot be eliminated by firmware updates, and hardware Trojan horses are therefore more detrimental to the computer system. Hardware Trojan horses are mainly used by hackers to add functions that are not needed in the chip. Mainly the attacker aims to steal our available resources from our system. There HT is classified into several types. In literature, researchers classified the HT detection classified as implicit and explicit payload based on Trojan performance. Once they activated. According to the invisibility of hardware Trojan horses and their possible impact, different methods of Trojan horses have been proposed. At the same time, those Trojans were injected at the (RTL)-register transfer level and they can inject via dopant manipulation.

During the Trojan activation time that test technology could not detect malicious logic before, in recent years, the purposeful hardware Trojan horse identification techniques and truthful integrated circuit design have been refined. The HT identification and precaution techniques were discussed, it's classified into four types: (i) enhanced functional testing; (ii) side-channel fingerprinting; (iii) prevention of Trojan horses; (iv) circuit hardening. Enhanced functional testing techniques are the key term of HTs usually trust in events that are uncommonly activated. So, we need to consider uncommonly activated events on test evaluation to activate the Trojan horse program meanwhile the test evaluation phase or verify every uncommonly activated event in the gate-level netlist to detect the suspicious areas that can be used as activation. The limitation of this pre-deployment technology is there is no standard definition of rare events, thereby eliminating the huge gap between the standard test mode and the rare event mode. Side-channel fingerprint recognition is one of the required solutions. However, the hardware Trojan cannot be easily triggered during the test phase and can be avoided by functional testing. The inserted Trojan must change the parameter configuration file of the polluted circuit. This model's accuracy depends on the ability to distinguish the side signals from circuits affected with Trojan from circuits without Trojan. Improved statistical methods can help create side-channel fingerprints by avoiding increased variation of process and measurement of the noise. The concept of Trojan horse program avoidance and circuit hardening is trying to change the shape of the circuit and some additional logic to avoid rare/suspicious functions. The above model is already integrated with other Trojan horse identification techniques to improve efficiency and reduce test costs.

5.1.2 Post-Deployment Hardware Trojan Detection

In the case where past Trojan horse detection methods have successfully detected specific hardware Trojan horses, the detection range is restricted as depend on oversimplified and few trials wrong expectations, also:

• Hardware Trojan horse programmer's uses classic, very simple circuit design shapes that may restrict its functions.

• Hardware Trojan horse program programmers try to fill a trivial on-chip location for cover up Trojan horse program configuration file from the overall side channel configuration file;

• The test under the circuit should have a golden model, it is used for identify changes in side channel profile; hackers mainly focus on discrete circuits, because analog-radio frequency circuits are more sensitive for make a changes.

The above assumptions were accepted by hardware security networks, so it has a set of rules and guidelines for improving the latest hardware Trojan identification/precaution techniques. Therefore, focused on post-deployment methods leveraging post-deployment side-channel fingerprinting and on-chip equivalence checking. The concept was that stealthy hardware, during testing time HT easily evades, if it's triggered, it contains a large impact on side-channel fingerprinting or circuit functionality. The first post-deployment trust evaluation structure has been proposed.

5.1.3 Formal Verification

In addition to circuit-level protection technology, formal verification is used for pre-silicon functional & safety verification. In entire formal techniques used for security verification, concept proof is an advisory answer that can provide too many levels of protection for the basic design. However, this technology also has the disadvantages of excessive calculation, tedious production systems, and lacks automation equipment. At the present stage, we can introduce a newly proposed hardware with proof (PCH) framework on the hardware IP core. It can further propose a fully formal protection technique based on the SAT solver for IP credibility verification.

• Proof-Carrying Hardware is a technology to ensure the reliability of hardware. The PCH model is activated by carrying a proof code (PCC), it was designed by G. Necula. Use of the PCC mechanism, suspicious software program builders/companies can prove their software code. On the certification process, software program builders and companies will extend the security certification through the protection criteria provided by the software customer. Then, most vendors provide users with a PCC digital file that contains a formal proof of production attributes encoded in the executable code of the software.

Consumers will quickly verify the PCC binary records in the evidence checker, thus being

confident in the protection of the software code. This technology is very efficient in reducing customer confirmation time, leading to unique packaging. Regardless of the benefits, the percentage method requires a huge trusted computing base (TCB). This shortcoming of PCC is conquered by the base percentage (FPCC), which makes Use basic logic on the specific operational semantics of different conference language commands, and implement the logic in the logic framework. However, the FPCC method greatly speeds up the development of evidence. Eventually, the certified assembly programming (CAP) language was designed, which used Hoare's common sense style reasoning for proof creation. The key concept of the PCC framework, the authors in designed preliminary carrier hardware (PCH) for dynamically reconfigurable hardware platforms. The important concept is to utilize the runtime combinatorial equivalence check between the initial design specification, design implementation on the reconfigurable system. Even if the technology is named PCH, it is based on a fully combined equivalence check of the SAT solver. The most effective exception is the resolution proof trace as a proof of functional equivalence. In this framework, the implemented security policy has nothing to do with security attributes. As an alternative, the scope of protection agreed by each bitstream company and IP client is to ensure that every aspect follows the same bitstream format, a joint common form for symbolic combination functions, and a propositional calculus for evidence generation and verification. Then a more specific creation of a framework based on the SAT solver and completely based on the proof on the configurable platform was delivered. In addition to the experimental setup, hardware considerations and dangerous versions.

• The solver is usually a proof maintain/ carrying technology, and is also used for security verification in register transfer-level code. For example, a four-step technique was proposed to remove and discover suspicious logic in third-party IP. In the first step, the use of function vectors created by sequential automatic test pattern generation (ATPG) eliminates easily identifiable signals. In a subsequent step, signals that are difficult to excite and/or propagate are indicated by using full scan N to detect ATPG. In order to narrow the scope of suspicious alarms and identify actual doors associated with Trojan horses, in the last step, it has been decided to use clusters of not testable doors within the circuit to use area isolation technology on the list of suspicious signals. some other multi-level methods were proposed to protect against announcement-based verification, code coverage analysis, redundant circuit deletion, equivalent analysis, and the use of sequential ATPG to handle suspicious signals. The technology is built on RS232 circuits, and its performance in identifying Trojan signals is between 67.7% and 100%.

5.1.4 Counterfeiting Prevention and IC Protection

The integrated circuit supply chain is mainly affected by preservation hazard from many factors, including IP stealing, IC duplicating, hardware backdoors and counterfeit integrated circuits. Various strategies were introduced for identify those assaults. Similarly, after knowing the seriousness of counterfeit chips in key structures, hardware security researchers began to study the field, hoping to find not original or not legally noted integrated circuits earlier than deployment. In statistical evaluation and device understanding concepts, it is used to restore

integrated circuits identification.

The aging sensor on the chip is a special solution for fake integrated circuits separation .One more growing problem is the overproduction of integrated circuits and IP stealing, which are usually caused by loss of supervision, and the direct participation at manufacturing later the style is transferred to fabrication area. And there is no viable result to decide whether the workroom is creating exactly the customer demanded, and whether it has overproduced chips. To solve this trouble, the concept of production reduction was changed. This enables IP vendors to rely on remotely producing products that are now unable to provide overall design statistics. This approach, the front end of the advanced production line (FEOAPL) is required to be manufactured in a remote place, but the back end (BEOAPL) of the production line is added to the domestic foundry so that the foreign foundry can only check part of the design information.

With the development of the design field, one of the greatest values of integrated circuits style is the development of the process and the refinement of the reverse engineering. Circuit design and the security of intellectual properties cores are getting more and more attention in integrated circuits delivery supply safety. This device allows reverse engineering of circuits and intermediate intellectual property replication is gaining a higher standard. Design confusion and logical encryption are different solutions, which can save the hacker from the problem of convulsing/replicating the circuit style without confusion the key. At the same time, the model has proven to be powerful assaults (intellectual property staling will only occur if the hacker understands the netlist and key), and the overall completion budget and format re-styling are severe. For example, sense the fault encryption technology based on failure analysis can increase energy dispersion with up to 188% of energy, delay energy consumption with up to 284% of delay, and can be invested as fast as 242% (assuming ISCAS takes a look at the bench).

Side channel assessment and fault insertion are other important problem to hardware security, especially for circuits that contain oversensitive data. Without physical intervention, the hacker could obtain better domestic indicators. These indicators can be analysed by static/differential analysis of aspects, including timing, energy intake and EM radiation. In addition to passive channel analysis, encryption circuits are also susceptible for complete fault insertion based on intensity supply. To counter these attacks, the researchers developed many good judgment circuits and on integrated circuits sensors to stabilize multi-faceted channel signals and encounter abnormal signals. Moreover, despite the assist of style enlargement techniques, and actual corrective measure bring large overall activities outlay.

5.1.5 Physically-Unclonable Functions

Orthogonal to hardware area protection is the improvement of safety essentials, which have been studied because they have higher accuracy, lower cost differentiated with software programs. The main prototype is the physically unclonable function (PUF), that uses tool not recognizes the generated by method changes to generate specific identifications for each chip and regular risk

response pairs. Although there are many factors to analyse physically unclonable function design, the main factors are,

• Randomness. It represents a dimension that shows how to give randomness of response to any input style.

• Expertise. This is another way to shows the robustness of the physically unclonable function design underneath variable climate conditions and/or noise.

• Excellent safety. Security items calculate the resilience of the physically unclonable function design underneath assaults.

Different assault models including device understanding and equipment modelling were improved, based upon the physically uncloned function response, we can break that physically unclonable function design. A lot of concepts were discussed for enhance the measure of error adjusting algorithms. The evaluations of physically unclonable function were discussed in various reviews.

5.2 Power Analysis for Different Benchmark

Detectable Trojan Size	Trojan Localization	Area overhead	Power	Reference
0.013-0.83%	Yes	0	Static Power	[40]
0.40%	Yes	0	Static Power	[39]
1.6-2.3%	Yes	0	Transient Current	[38]
0.09-2.61%	Yes	0	Transient Current	[37]
0.10%	Yes	0	Transient Current	[36]

Figure 5.2: POWER VS. TROJAN SIZE IN DIFFERENT BENCHMARK

Hao Xue, team members demonstrated that the detectable Trojan size is 0.013% of the host circuitry. They used ISCAS benchmarks for evaluation of efficiency. L. Zhang et al showed HT will contribute to circuit overall leakage/static power. These synthesized designs and then embed in multiple ISCAS85 benchmarks using a 65nm technology library, and perform a comprehensive power and area characterization. Y. Zheng et al explained The proof-of-concept based implementation of a black hat HLS framework it can introduce a variety of hardware Trojans that are challenging to detect and its show that a black-hat HLS tool can be successfully used to

maliciously alter electronic circuits to add latency, drain energy, or undermine the security of cryptographic hardware Cores. M. Banga et al demonstrated The generic formal framework that can analyze vulnerabilities in the circuits against functional and parametric HT.it can validate dynamic power, leakage power and path delay model. X. Wang et al proposed a threshold voltage-triggered HT it operates in a threshold voltage of 0.45V to 0.998V, consuming ultra-low power (10.5nW), and remaining stealthy with an area overhead as low as 1.5% for a 28 nm technology node [36].

Table results conclude that Hao Xue, team members demonstrated a better detectable Trojan size value of 0.013- 0.83% with static power analysis.

Chapter 6

ADVANTAGES, LIMITATIONS, AND APPLICATIONS

6.1 Advantages

- PUFs are easy to integrate into the design of a hardware system, as they do not require additional components or modification of the existing design.
- PUFs are resistant to duplication and reverse engineering, as they are based on intrinsic physical properties of the hardware, such as manufacturing variations or environmental conditions.
- PUFs can provide secure and unique identification or authentication of hardware devices, which is important for many applications, such as in the Internet of Things (IoT) or in secure communication protocols.
- PUFs can provide secure identification or authentication of hardware devices, making them ideal for applications that require secure communication protocols.
- PUFs are cost-effective, as they do not require additional hardware components, making them suitable for low-cost applications.
- PUFs can provide enhanced security for applications that require secure communication protocols, such as military and financial transactions.
- PUFs can provide tamper-proofing capabilities, making them ideal for applications that require secure storage of sensitive data.

6.2 Limitations

- PUFs can suffer from reliability issues, as the physical properties they rely on may change over time or due to environmental factors, such as temperature or aging.
- PUFs may be vulnerable to attacks that exploit their physical properties, such as laser attacks or voltage glitching.
- PUFs may introduce additional overhead and complexity to the design of a hardware system, such as in terms of testing or calibration.
- PUFs may not be suitable for applications that require high-speed data processing, as they may introduce additional latency.
- PUFs may have limited scalability for large-scale hardware systems due to the additional overhead they introduce.

6.3 Applications

- PUFs can be used for secure key generation and storage, where the unique physical properties of the PUF can be used to generate cryptographic keys that are resistant to attacks.
- PUFs can be used for secure authentication of hardware devices, where the unique physical properties of the PUF can be used to verify the identity of a device.
- PUFs can be used for secure anti-counterfeiting measures, where the unique physical properties of the PUF can be used to verify the authenticity of a hardware device or product.
- PUFs can be used for secure device pairing and communication in wireless sensor networks, where the unique physical properties of the PUF can be used to establish a secure communication channel.
- PUFs can be used for secure bootstrapping and secure firmware updates, where the unique physical properties of the PUF can be used to authenticate the firmware and prevent unauthorized access.
- PUFs can be used for secure identity management, where the unique physical properties of the PUF can be used to authenticate and manage user identities in secure systems.

Chapter 7

CONCLUSION

Based on the extensive research conducted on the topic of ”Defending against Hardware Trojans: the role of Physical unclonable functions,” it can be concluded that hardware Trojans pose a serious threat to modern computer systems, and the use of physical unclonable functions (PUFs) is a promising way to defend against them. PUFs are a type of hardware security mechanism that leverages the unique physical properties of a device to generate cryptographic keys, making it very difficult for attackers to tamper with or clone the device. There are different types of PUFs, such as silicon-based, ring oscillator-based, and magnetic PUFs, each with its own advantages and disadvantages. While PUFs may not offer an absolute solution to the problem of hardware Trojans, they are a crucial tool in the security arsenal of hardware designers and system architects. As the threat of hardware Trojans continues to escalate, PUFs and other hardware security mechanisms are expected to become increasingly vital in securing critical systems that support modern society. Thus, it is important to continue research and development in this field to enhance the security of our digital infrastructure.