Robust Solutions for Enabling Trust in Digital Circuits

by

Yuqiao Zhang

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
August 6, 2022

Keywords: Recycled
IC Detection, IP Piracy, Logic Locking, Supply Chain, Magnetic Skyrmion, IC Camouflaging.

Approved by

Ujjwal Guin, Chair, Assistant Professor, Electrical and Computer Engineering
Vishwani D. Agrawal, Professor Emeritus, Electrical and Computer Engineering
Mehdi Sadi, Assistant Professor, Electrical and Computer Engineering
Spencer Millican, Assistant Professor, Electrical and Computer Engineering
Shubhra (Santu) Karmakar, Reader, Asssitant Professor, Computer Science and Software Eng.

Abstract

The continuous emergence of counterfeit integrated circuits (ICs) in the electronics supply chain requires immediate solutions since they pose serious threats to our critical infrastructures due to their inferior quality. Information Handling Services Inc. reported that counterfeit ICs represent a potential annual risk of $169 billion to the global electronics supply chain and have continued to increase in recent years. These counterfeit ICs can be categorized into seven distinct types: recycled, remarked, defective/out-of-spec, overproduced, cloned, forged documentation, and tampered. It is reported that recycled ICs account for almost 80% of all reported counterfeit incidents. The rise of these recycled ICs in critical infrastructures can cause major concerns to the government and industry because these chips exhibit lower performance and have a shorter remaining lifespan. In addition, high temperatures, followed by sanding, repackaging, and remarking in the crude recycling process could potentially create extra defects in the ICs. The illegal recycling process may also create latent defects that can pass initial acceptance testing by original equipment manufacturers (OEM) but are susceptible to failure in the field. Furthermore, the detection of these ICs becomes extremely challenging when they are already circulating in the supply chain. Generally, it is necessary to power up a chip at a distributor's site to measure different electrical parameters to verify whether it was pre-owned or not. However, this can be challenging as many distributors may not be equipped with proper test infrastructures. Moreover, the reliability of authentic chips may be reduced if they were removed from their packaging boxes for testing purposes.

In parallel, due to the globalization in the semiconductor industry, the cost of maintaining a foundry is enormous. Hence, most integrated circuit IC design houses have become fabless. However, the security issue of intellectual property (IP) becomes another concern. Typically, a design house acquires multiple third-party cores for a system on a chip (SoC) and sends a contract to a foundry/fab for manufacturing and test. The horizontal integration of semiconductor design, manufacturing, and test causes a significant increase in potential harmful threats. The most notable ones are overproduction or counterfeiting of ICs, piracy of IP, and insertion of hardware Trojans

(HT) . To address these threats, researchers have proposed solutions that include hardware metering, logic locking, IP watermarking, and split manufacturing. Logic locking is the most widely accepted and design-for-trust (DFT) technique to prevent those threats from untrusted manufacturing. It hides the circuit's inner details by incorporating key gates in the original circuit resulting in a key-dependent locked counterpart. The resultant locked circuit functions correctly once the secret key is programmed into its tamper-proof memory. The inserted logic can be commonly categorized based on key-insertion strategy and can be described as: (i) XOR/XNOR-based, (ii) MUX-based, (iii) Look-up Table (LUT)-based, and (iv) state-space based. However, existing logic locking techniques can be rendered ineffective using the state-of-art methods that include Boolean satisfiability (SAT)-based attacks, probing, and tampering attacks. One can obtain the secret key from a functional chip and then unlock any number of locked ICs as the key value remains the same for every chip. Subsequently, different countermeasures were also proposed to increase the effort of launching these powerful logic-based attacks. Reverse engineering (RE) is another powerful attack for IP theft. It can be used by an adversary for the illegal reconstruction of the gate-level design. As a result, an adversary can clone an entire chip, pirate the extracted netlist, or insert a hardware Trojan. IC camouflaging can be an effective technique to prevent RE so that an adversary cannot obtain the inner details of a circuit.

In this dissertation, we developed a robust and low-cost solution for enabling the traceability of an IC in the supply chain. Our proposed solution builds a chain of trust among the manufacturer, distributors, and system integrator (end-user) by enabling end-to-end traceability from manufacturing to system integration and providing protection against IC recycling. The proposed solution utilizes a small passive radio-frequency identification (RFID) tag, which needs to be placed on the package. Any authorized entity in the supply chain can verify the authenticity of a chip using a commercial RFID reader. Once the system integrator verifies the signature from all the previous stages, final verification will be performed to detect the prior usage of an IC. The frequency of the IC will be measured again and compared with the stored value provided by the manufacturer. A mismatch will indicate that a recycled IC has been detected. Otherwise, it can be considered a brand-new product.

We also propose a new oracle-less logic locking attack based on self-referencing to determine the secret key. We denote our proposed attack as *TGA*: *T*opology-*G*uided *A*ttack on logic locked

circuits. Since the entire circuit topology is built from basic Boolean functions that are repeated multiple times (denote as unit function (UF)) in the design, it is possible for an adversary to determine the secret key by comparing the locked instances of these functions with the unlocked ones in the entire netlist. The secret key can be estimated efficiently even for the circuits in which the SAT attack has failed, e.g., *c6288* circuit. In addition, an adversary can unlock any netlist using our proposed attack without waiting for a working chip to be available in the market or with no scan access. We also proposed a countermeasure to prevent this attack. If the SoC designer locks all the repeated unit functions, the attack on the locked design will become ineffective as it cannot make a key prediction without self-referencing the unlocked unit function.

Spintronic devices offer a feasible choice for post-Moore devices, and magnetic skyrmion-based design becomes a possible candidate for implementing different logic designs and non-volatile memories. Reverse engineering is an advanced tool that can be exploited by the attack to recover the gate-netlist of an IP. IC camouflaging can prevent this attack, and various solutions are proposed for protecting traditional CMOS-based technologies. However, there is no research on camouflaging for skyrmion-based circuits. To solve this concern, we have proposed several novel skyrmion-based gate designs for implementing IC camouflaging on the skyrmion-based circuits. To the best of our knowledge, we are the first to propose camouflaged skyrmion (denoted as CamSkyGate: Camouflaged Skyrmion-based Logic Gates) gates to prevent an adversary from performing reverse engineering on a skyrmion-based circuit. The function of a camouflaged gate is determined by doping technology. The attacker cannot identify the specific gate type for the designed camouflaged gate since the layout is identical and symmetric even with different functions. The $\boldsymbol{mumax^3}$ simulator is used to exhaustively simulate all gates with different input combinations. We have also evaluated the security of the proposed camouflaged designs using SAT attacks. We show that the same security from the traditional CMOS-based camouflaged circuits can be retained.

This dissertation provides a comprehensive overview of different existing problems and solutions. To combat the recycled ICs in the supply chain, We have proposed an RFID-based system to enable traceability and trust among the manufacturer, distributor, and system integrator. Any entity in the supply chain is able to verify the authenticity of a chip using a commercial RFID reader. On the logic locking side, we have found the vulnerability of the existing logic locking

schemes and proposed an oracle-less topology-based attack. A countermeasure is also discussed to prevent self-referencing. To explore the security aspect of magnetic skyrmion circuit design, we propose the CamSkyGate design to prevent reverse engineering. An adversary cannot extract the full gate-level netlist by performing reverse engineering. The SAT-based security evaluation is also performed, where our proposed design can provide the same level of protection with a smaller key size than the CMOS-based camouflaged counterpart.

Acknowledgments

I would like to express my most heartfelt gratitude to Dr. Ujjwal Guin, my graduate advisor, for his encouragement and guidance during my study and life at Auburn University. His encouragement and guidance paved the way for my successful research projects and thesis completion. I would like to express my gratitude to Dr. Vishwani Agrawal for his experience and insight into research. I would like to thank Dr. Mehdi Sadi for helping me have a comprehensive understanding and experience in CAD design and related knowledge. Also, special thanks to Dr. Spencer Millican for his great help and efforts. I would also like to thank Dr. Karmaker for being my university reader and providing me with valuable comments and kind support. I want to extend my gratitude to the committee members again for their time, support, and advice toward my research and thesis preparation and implementation. Thanks to all of my labmates and colleagues for the valuable information I acquired during my course and study work. The lab exercises have taught me a lot about my field of study, and I owe much gratitude to them. Finally, I would like to thank my parents and friends for their support during my study and life at Auburn University.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

CRC	Cyclic Redundancy Check
DFT	Design-for-Trust
DIP	Distinguishing Input Pattern
DMI	Dzyaloshinskii–Moriya Interaction
ECID	Electronic Chip ID
EEPROM	Electrically Erasable Programmable Read-Only Memory
FF	Flip Flop
HT	Hardware Trojans
IP	Intellectual property
LLG	Landau–Lifshitz– Gilbert–Slonczewski
MTJ	Magnetic Tunnel Junction
NBTI	Begative Bias Temperature Instability
NVM	Non-volatile Memory
OEM	Original Equipment Manufacturers
PCB	printed circuit board
PMA	Perpendicular Magnetic Anisotropy

RE Reverse Engineering

RFID Radio-Frequency Identification

RO Ring Oscillator

SAT Boolean Satisfiability

SEM Scanning Electron Microscopy

SLL Secure Logic Locking

SoC System On Chip

TMR Tunneling Magnetoresistance

UF Unit Function

Chapter 1

Introduction

1.1 Motivation

The security and trust of the electronics supply chain attract more attention and concerns since the number of counterfeit integrated circuits (ICs) has increased significantly. The increased source of e-waste can help the counterfeiters build up an extremely large supply of counterfeit components to make more profits. The degraded performance and shorted lifetime compared with authentic chips pose serious threats to the critical infrastructures [2–5]. A report has indicated that a potential annual risk of $169 billion is presented by counterfeit ICs to the global electronics supply chain and have continued to increase [6]. Recycled, remarked, defective/out-of-spec, overproduced, cloned, forged documentation, and tampered ICs are the seven main types of different counterfeit ICs. The recycled ICs contribute to more than 80% of counterfeit incidents and need immediate solutions for their detection. These chips can cause major concerns to the government and industry because of lower performance and shorter remaining useful life. Generally, these components are taken from used printed circuit boards (PCBs), repackaged, remarked, and then sold in the market as new. The high temperatures during the crude recycling process may also create extra defects in the ICs. When the ICs circulate in the supply chain, it becomes extremely challenging to detect them. It is mandatory to power up a chip and collect critical electrical parameters to perform the recycling detection. However, without the proper test infrastructure, it is extremely difficult to perform the test and verification at a distributor's site. Furthermore, if the ICs were removed from their packaging for testing purposes, the reliability of authentic chips may be compromised due to electrostatic discharge (ESD) or humidity.

With the trend of globalization in the semiconductor industry and the high cost of maintaining a foundry, most IC design houses have adopted the fabless model. Generally, a design house first buys the IPs cores from multiple different third-party IP vendors for developing a system on a chip (SoC). A contract is then sent to a foundry/fab for the manufacturing of chips. However, this current well-accepted mode has caused a significant increase in potential harmful threats. These threats can be the overproduction of ICs [7–11], piracy of intellectual properties (IPs) [4, 11–13], and tampering with hardware Trojans [4, 11, 14–18]. Researchers have proposed different solutions in recent decades to prevent these threats. The countermeasures include hardware metering, logic locking, watermarking, and split manufacturing. Logic locking attracts the most attention and is the most widely accepted scheme for preventing both IC overproduction and IP piracy. The basic idea of logic locking is to lock the function of a circuit by a secret key into tamper-proof memory on the chip. When the chips are fabricated, the SoC designer needs to program the key and activate the chip. Only the correct key makes the IC fully functional. The inserted logic can be commonly categorized based on different key-insertion strategy: (i) XOR/XNOR-based [8, 19–22], (ii) MUX-based [23–25], (iii) LUT-based [26–28], and (iv) state-space based [29]. Several attacks, such as Boolean satisfiability (SAT)-based attacks [30–38], probing [39], fault-injection [40–42] and tampering attacks [43, 44, 44] can compromise the security of the existing logic locking schemes. The correct secret key can be obtained by performing these attacks on an activated chip from the market. Therefore, all ICs of the same type can be compromised, and the hardware security objective against IC overproduction and IP piracy will not be fulfilled. To extract the netlist, which is required for the attacks mentioned above, reverse engineering [45, 46] needs to be performed. The full gate-level netlist and function can be illegally constructed by an attacker. As a result, an adversary can clone an entire chip and pirate the extracted netlist to make profits without spending the cost and time of Research & Development. IC camouflaging is an effective technique to prevent RE so that an adversary cannot extract the inner details of a circuit [47–49].

1.1.1 Prior Work

Detection and Avoidance of Recycled ICs

The detection and avoidance of recycled ICs, a major source of counterfeit incidents, requires innovative solutions. Different solutions have been proposed to detect recycled ICs over the years [50–56]. First, there are several standards (e.g., AS6171, AS5553, AS6081, CCAP-101, and IDEA-STD-1010) in practice, which recommend different physical and electrical tests for the detection purpose [57–61]. The goal of these tests is to identify defects and anomalies present in recycled chips. However, excessive test times, test costs, low detection capability, and lack of automation make the detection of recycled ICs extremely challenging. Second, different solutions have been proposed, which are based on statistical data analysis [50–55]. However, these solutions provide limited accuracy when the chips are used for a short period. In addition, these schemes require authentic samples to train the model, which may be difficult to acquire from the market. Third, when the ICs are aged, a result of negative bias temperature instability (NBTI) can be caused. NBTI happens whenever a PMOS transistor is stressed with negative gate voltages and elevated temperatures due to the generation of interface traps at the Si/SiO2 interface. The aging of an IC will cause an increment in the threshold voltage, and it decreases the drain current of the MOSFET. This feature can be exploited to detect counterfeit IC in the market. For example, Authors in [62] propose a lightweight anti-counterfeiting technique for combating die and IC recycling (CDIR) that is based on Schmitt-trigger Ring Oscillator (STRO) sensors. A modified version is introduced in [63], which is based on ring-oscillator (RO) sensors. Two groups of RO sensors are placed to determine the counterfeit ICs. One group of RO is stressed as normal, while the other is not stressed as a reference RO. A counter with a timer is also implemented to measure the oscillation frequency of each RO group. The frequency difference between the two groups will indicate the prior usage of an IC. Different other on-chip sensor-based solution are also proposed to detect the recycled ICs [64–69]. Unfortunately, these solutions are sensitive to process variations, and lower accuracy might be caused when the process variations outpace aging degradation. To address this limitation, The solutions proposed in [1, 70] are resistant to process variation, and they can detect recycled ICs accurately even within a small

amount of time. Fourth, a zero-cost approach using embedded SRAM is proposed in [55]. The start-up of the SRAM will output a stable value for fresh IC but will become unstable after aging. Authors in [56] provide a statistical analysis technique that relies on the circuit delay of multiple paths. It has been demonstrated that delays between two or more paths in the circuit can be used as an age indicator. However, these solutions can only provide limited accuracy when the chips are used for a short period. In addition, this kind of detection requires obtaining reference data from a brand new golden chip on the market which affects its implementation. To address the limitation, the work proposed in [71] can detect the prior usage of a chip efficiently by observing the power up state of the flip flops (FF) . Finally, other different defections are also proposed for determining the recycled ICs. Image processing is a possible way used to detect recycled ICs [72, 73]. It is a physical inspection technique that involves investigating various external/internal properties such as texture, indents, marks, imperfections, or the package of an IC. However, their effectiveness also depends on the availability of authentic chips. DNA markings proposed in [74] can provide traceability for electronic parts and avoid the recycled ICs from another perspective. Unfortunately, it has limitations in practice because of the complex authentication process, excessive implementation, and test cost [75].

Logic Locking

Logic locking is widely accepted as a protection technology against IP Piracy and IC Overproduction. The original design could be protected by adding extra key gates. Once a correct key is programmed, an IC will work with full functionality. Random logic locking (RLL) [76], strong logic locking (SLL) [77], and fault-based logic locking (FLL) [23] were proposed. The basic idea is to insert XOR/XNOR gates in the gate-level netlist based on different insertion strategies. The RLL scheme was vulnerable to sensitization attacks in which the correct key bits can be propagated to the primary output by applying different input patterns [78]. Compared with a random key gate location selection, SLL will form an interference graph of key gates for examination, and the best candidates will be selected for key gate insertion. Unfortunately, these solutions were broken by performing SAT-based attacks [30]. The SAT-based attacks can find the correct key by ruling out incorrect ones iteratively with of help of applying distinguishing input patterns (DIP) .

SARLock was proposed in [79], where a small comparator circuit is added to the design. Anti-SAT, proposed in [80], inserts a relatively light-weight circuit block (two complementary functions). The main idea of these countermeasures is to make sure that each DIP can determine only one wrong key. Even though these two countermeasures can prevent the SAT-based attacks efficiently, they are broken by performing other attacks. The removal attack proposed in [81] can bypass the lock circuitry to retrieve the original circuit and remove dependence on key values, while the work proposed in [82] can identify the locked block of Anti-SAT efficiently. To improve the security, TTLock [83] was proposed as an improved version of SARLock. Also, the authors in [84] propose the stripped function logic locking (SFLL). For each incorrect key, there exists a very limited number of input patterns plus one extra input pattern (caused by the stripped function) that corrupts the POs. However, these improved countermeasures are compromised by functional analysis (FALL) attack [85]. A hardware Trojan-based attack (e.g., TALL [43]) is also powerful to thwart the above-mentioned techniques.

Beyond those countermeasures focusing on combinational part locking at the gate level, the sequential logic locking (e.g., HARPOON locks finite state machines [29]) and logic locking at high levels such as TAO [86] in high-level synthesis(HLS) or HLock [87] at the register-transfer level (RTL) were also proposed to lock a design.

Camouflaging

The RE has become one of the primary security concerns to the industry and government [88]. Even though it is recognized worldwide with many noble purposes [2], such as failure analysis, defect identification, hardware Trojan detection, or detection of counterfeit products, it can also be exploited by the adversary with different malicious purposes such as cloning and piracy. Generally, the RE-based attacks recover the full layout through scanning electron microscopy (SEM) imaging process [89].

IC camouflaging has become one of the popular research areas in combating RE. The researchers have proposed different solutions to secure the design through IC camouflaging. In [89], the authors develop a secure cell library design through an AND-tree structure, and a combined strategy is also adopted to reduce the cost. In [90], a dummy contact-based is proposed two control adjacent layers.

The transistors are in the ON or OFF state to prevent RE by changing the dopant polarity of the post-silicon state. Threshold-dependent implementation is considered as another direction [48, 91, 92]. Different gate functionality can be implemented based on different defined threshold voltages. The same layout will be recovered when RE is performed as it cannot help the attacker to determine the gate type. The authors in [93] use a pull-down resistor to control the current flow and decrease the average power and delay. This will cause a smaller voltage swing from logic-1 to logic-0. A cyclic obfuscation under different cycle conditions proposed in [94] is proposed to increase the effort of RE with an exponentially increasing time. Another way of IC camouflaging is denoted as 'covert gate' that leverages doping and dummy contacts to design camouflaged cells [49]. Authors in [95] proposed an IC camouflaging scheme by toggling the output for one minterm of the perturbed function. A separate camouflaged block is necessary to restore the perturbed minterm. These proposed solutions can be applied to prevent the RE on traditional CMOS-based technology. However, there is hardly any related research on camouflaging emerging skyrmion-based technologies.

Design and Simulation of Skyrmion Gates

Due to the quantum-mechanical limit, the development of IC with CMOS technology is getting slower, and an alternative solution is urgent. Spintronic devices offer a feasible choice for post-Moore devices. Magnetic skyrmions are considered in new magnetic data-storage devices and logic applicants due to their small size and low critical current density. A skyrmion behaves like a stable pseudoparticle and can be generated by a localized spin-polarized current. The presence of a skyrmion can be represented as binary logic 1 while logic 0 indicates the absence of a single skyrmion, respectively. The Skyrmion logic gates design relies on the utilization of different effects for skyrmion motions, such as spin-orbit torque-induced skyrmion Hall effect [96, 97], skyrmion-edge repulsion [98], and spin-orbit torque-induced motion [99, 100]. To simulate the proposed design, $\mathbf{mumax^3}$ and $\mathbf{OOMMF}$ are two popular tools to evaluate the motion of skyrmions and the function of the gate design.

There are several different skyrmion-based logic gate designs. In [101], the authors have designed spin logic gates such as the AND and OR gates based on manipulations of skyrmions. The conversion of a skyrmion into another form of a skyrmion. In [102], the authors have proposed

not only the basic single gate design (AND, OR. BUFFER, and INVERTER) but also a complex Fredkin gate. A re-configurable skyrmion-based logic gate is proposed in [103] which can function between AND, OR, NAND, and NOR on a single skyrmion gate design. In addition to different logic gate designs, the work in [104, 105] also developed the fan-out structure for skyrmions and evaluated the testing performance on different function types of skyrmion-based logic gates, which helps create a reliable future of spintronic devices. Even though the advantages of minimal power consumption and small device size make skyrmion-based circuit becomes a competitive candidate beyond traditional CMOS technology.

1.2 Contributions

This dissertation proposes novel solutions to address the threats mentioned above. The contributions of this dissertation are summarized as follows:

- *Enabling supply chain traceability:* We address the current limitation of combating the counterfeit ICs in the supply chain, and make the following contributions:
  - We propose a solution to enable the traceability of ICs in the supply chain against recycling. The core of the proposed structure is to utilize a small RFID tag that contains a small non-volatile memory (NVM) to be placed on the package. The chip needs to be equipped with a ring oscillator (RO) and an electronic chip ID ($ECID$). We propose to store the registration data that consists of the RO frequency and the frequency measurement conditions in the tag. A digital signature, which is computed on the registration data (RD) and $ECID$, also needs to be stored in the tag to prevent tampering with the RD. Recycled ICs can be detected by comparing the RO frequencies stored in the tag with measured values from the chip.
  - The solution sets up a chain of trust among the distributors and empowers them to verify the identity of all prior distributors who have possessed the IC. We use the concept of blockchain, originated from the cryptocurrency system Bitcoin [106], to develop our proposed solution. We believe this is the first approach to enable traceability for chips using RFID while they travel through the supply chain. The end user can uniquely identify the complete route of a chip by verifying the RFID tag content. Any modification or tampering with the RFID tag data can easily be detected as they are protected using digital signatures.

- Our proposed solution enables a distributor to verify the authenticity of a chip without powering it up. This is a significant improvement compared with the traditional barcode-based tracking [107], which can easily be cloned or tampered. We believe that this is the first approach that enables verification at a distributor's site without powering up the chips.

- *Attacks and defenses on logic locking:* We investigate the vulnerabilities of the current logic locking designs, and an oracle-less topology-based attack is proposed with performance evaluation. An effective countermeasure is also provided. The contributions are summarized as follows:
  - *A novel oracle-less topology-guided attack on logic locking:* We proposed a topological function search attack that relies on identifying and searching the repeated functions in a netlist. We denote these basic functions as unit function *UF*, which are repeated multiple times in a circuit. If a key gate is placed in an instance of a repeated *UF* during the locking of a circuit, the original netlist can be recovered by searching the equivalent unit functions (*EUFs*), which are constructed with all hypotheses' key values. As the *UFs* are constructed in a few layers of gates, the number of key gates and key bits associated with a *UF* is limited, resulting in minimal *EUF* search combinations. The results in Table 3.1 show the efficiency of the proposed attack by recovering the majority of key bits correctly for ISCAS'85 [108] and ITC'99 [109] benchmark circuits locked with random logic locking (*RLL*) and secure logic locking (*SLL*). The effectiveness of our proposed *TGA* is also validated using locked benchmarks from TrustHub [110] (see Table 3.2). In contrast with the traditional oracle (unlocked chip) attacks, no oracle is required to launch our proposed attack.
  - *An efficient function search algorithm:* To perform the search, an efficient depth-first-search (*DFS*) based algorithm is developed to find the equivalent unit functions in a locked netlist. The complete netlist is first converted to a directed graph [111], where each gate in the netlist is represented as a vertex, and each wire is modeled as an edge. We demonstrate and implement a *DFS*-based *EUF* search algorithm to determine the correct value of a secret key. The average time to determine a secret key bit is in the order of seconds. As a result, a locked circuit can be broken in a few minutes when locked with a few hundred/thousand key gates.
  - *A countermeasure against the proposed TGA attack:* As the proposed attack recovers the original design by performing the *EUF* search on the netlist, it can be prevented if the

function search with hypothesis keys does not find results or produces contradictory results. This resiliency against the attack can be achieved by inserting the key gates in all the repeated instances of an *UF* as the adversary will not decide the actual value of the key bit by comparing it with its unlocked version. An *DFS*-based search algorithm is again exploited to identify all repeated and unique instances of a unit function. Note that the key length can be variable in a range instead of a fixed value, which can increase both the efficiency of the key insertion and the security of the locked design.

- *Secure IC camouflaging using magnetic skyrmions:* Since no secure camouflaging scheme has been proposed so far using spintronic devices, such as magnetic skyrmions-based circuits, we have the following contributions to enhance the robustness of this emerging technology and provide similar security compared with traditional CMOS technology [112]:
  - We propose novel CamSkyGate, Camouflaged Skyrmion-based logic Gates, for protecting ICs against RE. A camouflaged gate can be configured between AND and OR, and OR and buffer (BUF). A complex camouflaged gate that can be configured between a two-input AND with a dummy input, two-input OR with a dummy input, 2-to-1 MUX, and a BUF with two dummy inputs are also proposed. We present the layout for these different gates and perform micromagnetic simulations to verify the functionality of these gates.
  - Unlike the CMOS counterparts, the skyrmion-based gates that we propose are non-volatile, compact, and do not require extra components for camouflaging. The camouflaging can be realized by doping selected regions with different magnetic anisotropy.
  - We evaluate the security of our proposed design against the SAT attack. The experiment results show that our proposed design has a similar security level compared with the existing CMOS-based IC camouflaging scheme [49].

1.3 Organization of this Dissertation

The rest of this dissertation is organized as follows:

- Chapter 2 introduces our proposed solution for enabling end-to-end traceability against recycled ICs in the IC supply chain. A chain of trust among the manufacturer, distributors, and system integrator can be constructed. A small RFID tag needs to be placed on the package. Digital

signatures are exploited to protect the integrity of data. Each entity is able to interface with the RFID tag (e.g., read and write) through a commercial RFID reader. Once all the signatures are validated by the system integrator (end-user), it will perform the final recycled IC detection.

- Chapter 3 describes an oracle-less attack based on the self-referencing. The attack relies on identifying repeated functions for determining the value of a key bit. The proposed attack does not require any data from an unlocked chip, eliminating the need for an oracle. It compares the internal netlist to find the key. The proposed graph search algorithm efficiently finds a duplicate function of the locked part of the circuit. Our proposed attack correctly estimates a key bit very efficiently, and it only takes a few seconds to determine the key bit. We also present a solution to thwart the attack and make logic locking secure.
- Chapter 4 introduces our proposed skyrmion-based gate designs for IC camouflaging against reverse engineering. With different levels of doping, different Boolean functions can be implemented with an identical layout. The attacker cannot extract the specific gate type of the camouflaged cell. Thus, the full function of a camouflaged skyrmion-based circuit is secure against RE. The SAT-attack resilience is also discussed and evaluated in this chapter.
- Chapter 5 concludes the contributions of this dissertation, which also outlines the possible future work. Hardware security has attracted more attention and plays an important role in traditional CMOS technology and emerging technologies. We end this chapter with a list of some future research directions.

Chapter 2

End-to-End Traceability of ICs in Component Supply Chain for Fighting Against Recycling

2.1 Introduction

The continuous growth of ICs in the electronics supply chain calls for an immediate solution as they pose serious threats to our critical infrastructures due to their inferior quality. Information Handling Services Inc. reported that counterfeit ICs represent a potential annual risk of $169 billion to the global electronics supply chain, and this continues to increase in recent years [6]. These counterfeit ICs can be categorized into seven distinct types, such as recycled, remarked, defective/out-of-spec, overproduced, cloned, forged documentation, and tampered types. Among all these different counterfeit categories, recycled ICs account for almost 80% of all reported counterfeit incidents [4]. The deployment of these recycled chips in a critical infrastructure would be catastrophic as they exhibit lower performance and shorter remaining useful lifetime than a newly manufactured IC [3]. In addition, the crude recycling process that consists of removal of the ICs from printed circuit boards (PCB) under extremely high temperature, followed by sanding, repackaging, and remarking could potentially create many defects and anomalies [2, 4]. Moreover, the recycling process may also create latent defects that can pass initial acceptance testing by OEM but are susceptible to failure in the field [4].

The detection and avoidance approaches for recycled ICs are broadly classified into five different categories. First, there are several standards (e.g., AS6171, AS5553, AS6081, CCAP-101, and IDEA-STD-1010) in practice, which recommend different physical and electrical tests for the detection purpose [57–61]. The goal of these tests is to identify defects and anomalies present in recycled chips. However, excessive test times, test costs, low detection capability, and lack of

automation make the detection of recycled ICs extremely challenging. Second, different solutions have been proposed, which are based on statistical data analysis [50–55]. However, these solutions provide limited accuracy when the chips are used for a short period. In addition, these schemes require authentic samples to train the model, which may be difficult to acquire from the market. Third, on-chip sensors have been proposed as an alternative to the conventional test methods [64–69]. Unfortunately, these solutions can provide lower accuracy when the process variations outpace aging degradation. Fourth, image processing is also used to detect recycled ICs [72, 73]. However, their effectiveness depends on the availability of authentic chips. Finally, DNA markings are commercially available to provide traceability for electronic parts [74]. However, a complex authentication process, excessive implementation, and test cost have made its application limited in practice [75].

The solutions proposed in [1, 70] are robust against process variation, and they can detect recycled ICs very accurately even under minimal prior usage. Due to the complex nature of the semiconductor supply chain, electronics travel through many distributors (trusted and untrusted [57, 58]) before being deployed to a system. The only approach to determine whether a chip is recycled or not is by powering up a chip and measuring the ring oscillator frequency. However, this can be challenging to distributors as they may not have the proper test infrastructures. Moreover, accessing an individual chip (removing it from the packaging and then place in the tester) may create additional defects. *Thus, it is a mandatory requirement for a distributor to verify the authenticity of a chip without powering it up.*

2.2 Prior Process Variation Resilient Approach

The solution proposed in [1] utilizes a RO and a non-volatile memory, where – (i) the registration data (RD) that consists of the frequency (C_0) of an RO and the conditions (e.g., supply voltage (V_0), temperature (T_0), and duration (t_{D0}) for the frequency measurement), and (ii) a digital signature ($Sig(H_d)$) on data (d) that consists of RD and electronic chip ID ($ECID$) are stored. $ECID$ provides a unique identification to each chip. It generally includes the X-Y locations of a die in the wafer, lot information, wafer number, binning information, speed grade, etc., for traceability purposes [113]. This $ECID$ value can be accessed using an *ECIDCODE* instruction defined in Std 1149.1 [114]. Note that $ECID$ is the unique identification for a chip, not the RFID

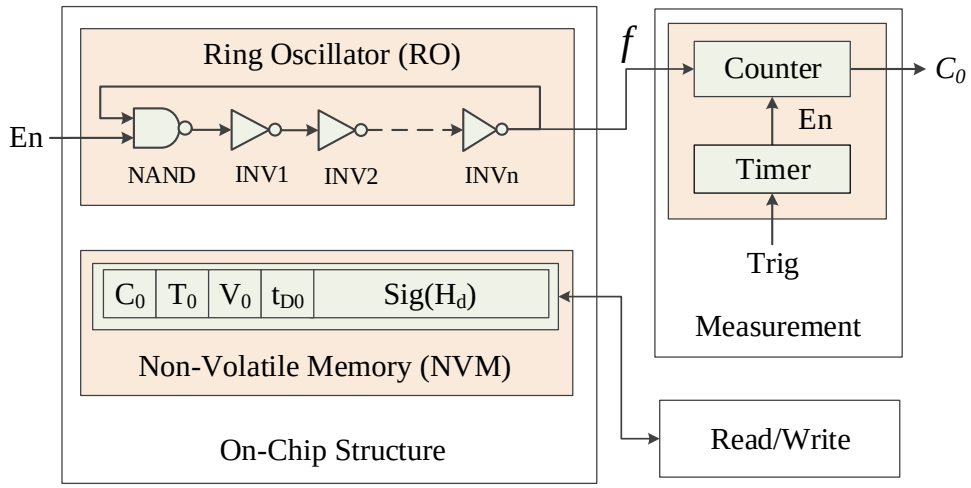

Figure 2.1: Prior process variation resilient on-chip structure for detecting recycled ICs [1].

device. The detection requires the verification of the signature to detect tampering with the NVM content and the comparison between the measured and stored frequencies. This approach helps to detect recycled ICs used as little as a day with a very low-cost measurement unit.

Figure 2.1 shows the structure proposed in [1] for detecting recycled ICs. It consists of an RO and an NVM. This RO can be selected from one of the process monitors [115–117] currently used in modern chips. The output of this RO can be made available using an existing primary output (PO) through multiplexing primarily to reduce the pin count. A counter and a timer are required to measure the RO frequency. One can also use the existing on-chip counter and timer for the frequency measurement. Test access port and boundary-scan architecture [114] can be used to access the NVM content. The solution requires generating the digital signature and then programming it into the NVM during the registration phase. The identity of the chip is verified during the authentication phase.

2.2.1 Registration Process

The registration phase starts after the chips are manufactured and tested for defects. Only the defect-free chips go through the registration process. During this phase, the frequency of the ring oscillator is measured by using a low-cost measurement unit. Next, the digital signature is constructed on the sensor data. Finally, the data will be programmed into an NVM. The steps are described as follows:

1. Ring oscillator data (RD) is constructed by concatenating counter value and measurement conditions.

$$RD = \{C_0||T_0||V_0||t_{D0}\}$$

2. Data (d) is constructed by concatenating RD and $ECID$.

$$d = \{RD||ECID\}$$

3. The digital signature ($Sig(H_d)$) is constructed on the hash of d with the OCM's private key. This secure private key is only available to the OCM.

$$\begin{aligned} H_d &= hash(d) \\ Sig(H_d) &= K^-(H_d) \end{aligned}$$

where, $hash()$, K^-, $K^-()$ represent a secure hash algorithm (SHA-2/SHA-3 [118]), private key, and the encryption function (RSA or ECC [119]), respectively.

4. The oscillator data RD, and the digital signature $Sig(H_d)$ are stored in the NVM of the chip.

2.2.2 Authentication Process

The authentication process can be described as the process of verifying the authenticity of a device. It can be straightforward and performed by the system integrator or end-user with a very low-cost measurement set-up, which has to be equipped with a counter and a timer (see details in [1]).

During this phase, it is necessary to read the $ECID$ and NVM content from the chip. The signature comparison is performed to verify the integrity of the NVM content. At the end of the authentication process, the age of the chip is determined by comparing the stored RO frequency with the measured RO frequency. The steps are described as follows:

1. The NVM content that consists of the ring oscillator data (RD) and digital signature ($Sig(H_d)$) of the chip under authentication, and the $ECID$ value is read. The data (d) is now constructed

by concatenating RD and $ECID$.

$$d = \{RD||ECID\}$$

2. A hash (H_d) is computed on d, and another hash (H_d^*) is recovered from the signature ($Sig(H_d)$).

$$H_d^* = K^+(Sig(H_d))$$

where, K^+ represents the public key.

3. The computed hash (H_d) and the recovered hash (H_d^*) are tested for any mismatch. Any mismatch indicates the tampering of the NVM content by an adversary, and the chip will be flagged as recycled.

4. If the hashes match perfectly, the measurement parameters during registration (T_0, V_0, and t_{D0}) are extracted from the ring oscillator data (RD).

5. The RO clock cycle count (C_0^*) is measured using parameters t_{D0}, T_0, and V_0.

6. The difference between the measured clock cycle count (C_0^*) and the registration clock cycle count (C_0) is calculated. If the difference is greater than the precision of the counter (measurement error), the chip will be identified as recycled.

2.3 Proposed Comprehensive Approach for Combating IC Recycling

The solution proposed in Figure 2.1 can detect recycled ICs accurately even though they have been used for a short period of time. However, it is necessary to power up the chip when an entity in the supply chain wants to verify whether it has been used before or not. Note that a chip travels through many distributors, some can be untrusted, before being deployed to a system. It can be challenging for many distributors to adopt the solution proposed in [1], as they may not have the proper test infrastructures. Besides, access to individual chips may be infeasible as unpackaging may create many defects and anomalies from improper handling.

Figure 2.2 shows our proposed design, where the chip is equipped with an RFID tag. We propose to move the on-chip NVM contents, such as the registration data, the signature, and other information (see Section 2.3.1) to an RFID tag. The die of a chip only contains the ring oscillator to determine the age, whereas, the RFID tag can be placed in the package during the packaging stage of the manufacturing process for enabling traceability of chips in the supply chain. Note that a similar effort to integrate RFID in the chip is ongoing by the Supply Chain Hardware Integrity for Electronics Defense (SHIELD) initiative by DARPA [120].

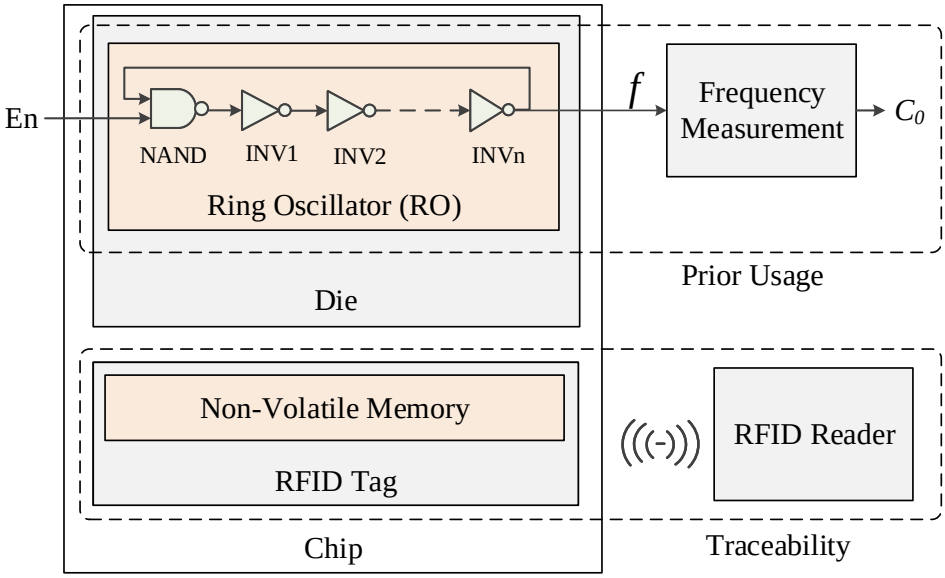

Figure 2.2: Proposed design for enabling traceability of ICs in the Supply Chain for combating against recycling.

In recent years, RFID tags have been widely used for traceability in the supply chain. There are two basic types of RFID tags in use: passive and active tags. For active RFID tags, the tag's lifetime may be limited by the energy stored in the integrated battery. On the other hand, passive tags are more popular due to their lower size, cost, and longer lifetime. As these tags do not require a battery, they can be small enough to put into a label attached to the product. Even though the RFID solution provides flexibility for device identification, its contents are vulnerable to unwanted modifications. Our solution provides protection against it as the contents are digitally signed.

2.3.1 Proposed Approach for Enabling Traceability of ICs

The traceability of a component in the supply chain can be achieved by creating a chain of trust among the manufacturer, distributors, and the user of these chips. We use the concepts of the blockchain, which was introduced in the Bitcoin cryptocurrency system by Satoshi Nakamoto in 2008 [106]. Bitcoin uses a hash-based block structure and a consensus algorithm, denoted as Proof-of-Work (PoW), to achieve decentralization. We do not need a consensus algorithm for traceability purposes as the endpoints of the component supply chain are trusted, and chips are the transactions. As a result, the concern of double-spending will never arise. In the supply chain, the manufacturers of chips are treated as trusted (see General Requirements of the Standard AS6171 [57]), as there is no motivation for a manufacturer to recycle chips and send them into the market as new. In addition, the end-users (e.g., Honeywell, NASA, Boeing, etc.) of the chips are also considered trusted as they are suffered from recycled chips. Thus, our objective here is to identify a recycled (used and old) chip that enters the supply chain through untrusted distributors.

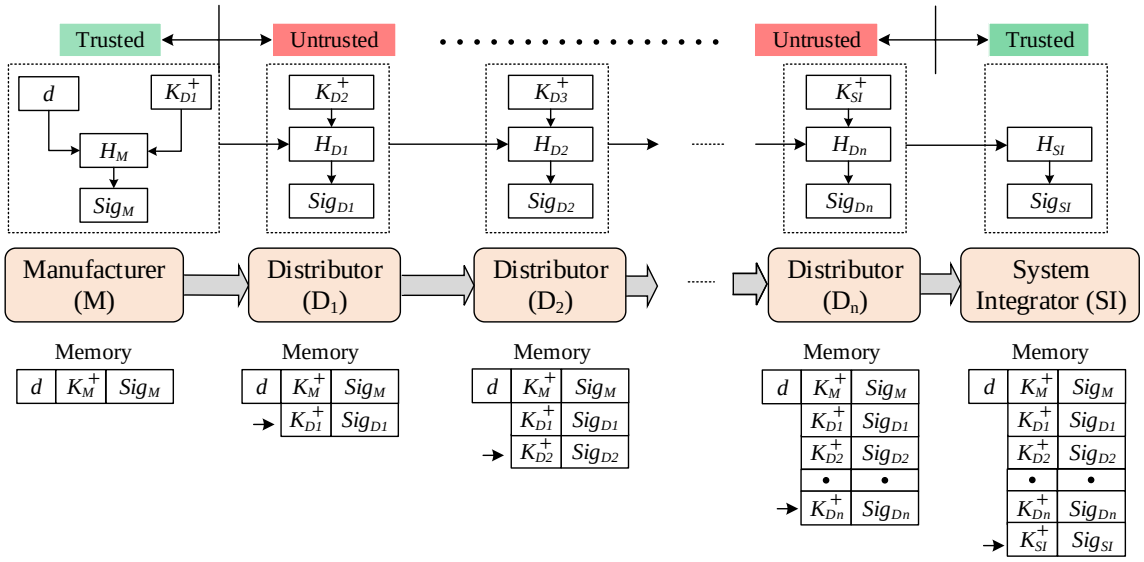

Figure 2.3: Proposed flow for enabling traceability of ICs in electronic component supply chain.

Figure 2.3 shows the proposed solution to enable traceability of chips while they travel through the electronics supply chain. First, the manufacturer reads the RO frequency (C_0) once the chip is free from manufacturing defects. The parameters during the measurement process (e.g., supply voltage (V_0), temperature (T_0), and duration (t_{D0})) are also recorded. The data (d) is constructed

by concatenating these parameters with the $ECID$, where $d = \{C_0||V_0||T_0||t_{D0}||ECID\}$. A cryptographically secure hash (H_M) is computed on d and the ID of the first distributor (e.g., public key of D_1, $K^+_{D_1}$). A digital signature (Sig_M) is then computed on H_M. The manufacturer updates the RFID tag memory with $\{d, K^+_M, Sig_M\}$ using a commercial RFID reader, and later ships the chip to the distributor, D_1. Second, the distributor D_1 first reads the RFID content using a commercially available RFID reader once it receives the chips from the manufacturer. It then verifies the integrity of the RFID content. If the verification passes, D_1 creates a hash (H_{D_1}) on the previous stage's hashes and signatures, and the next distributor's public ID (e.g., $K^+_{D_2}$). It then computes the signature (Sig_{D_1}) on H_{D_1}, updates the RFID with $K^+_{D_1}$, Sig_{D_1}, and sends the chip to the next distributor (D_2), which also performs the same steps as D_1. Finally, the system integrator (SI) verifies the entire RFID content and constructs the complete trace. Once the chip has been placed into a system, SI updates the RFID memory with its own signature to certify that it has been deployed.

Note that the IDs of different manufacturers, distributors, and system integrators (users) are required to be stored in a secure database and can be accessed through a trusted website such that one cannot tamper with these IDs. One can also store the certificates [121] in the RFID, however, it may require larger tag memory space.

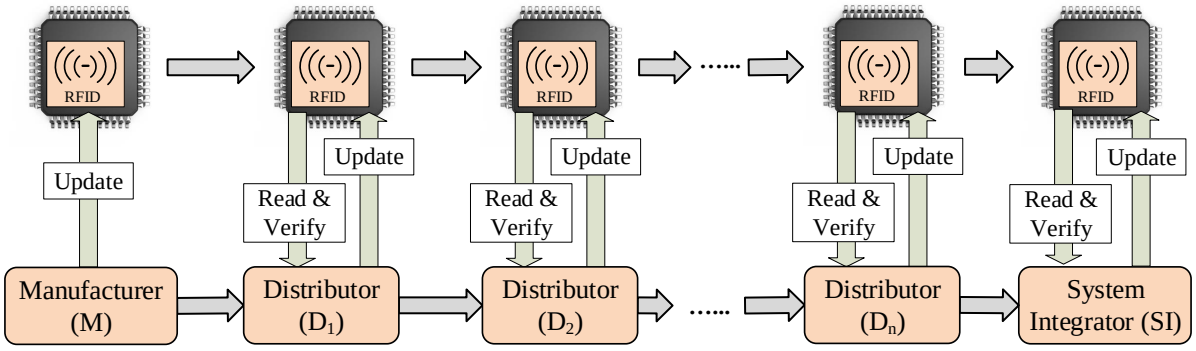

Figure 2.4: Verification and update process for the contents of an RFID tag placed in the package of a chip.

Figure 2.4 shows our proposed approach for enabling traceability in the supply chain. The proposed approach consists of three stages: (1) read RFID content, (2) verify RFID content, and (3) update RFID content. Note that the manufacturer only performs the update operation, as there are no prior entities in the supply chain and it is not required to verify RFID content.

Read RFID Content

The first step is to extract the information stored in the RFID tag. This can be performed through a commercial RFID reader without powering a chip.

Verify RFID Content

The distributor needs to perform signature verification for all prior stages starting from the OCM. Note that each row in the memory contains the public key (K_i^+) of the manufacturer (first row), the system integrator (last row), or the distributor (other rows), and the signature (Sig_i). The verification can be performed as follows:

- Step 1: The content in the 1^{st} row of the RFID memory needs to be read first by i^{th} distributor (D_i), which was created by the OCM. A hash H_M is computed based on d and the public key of distributor 1.

$$H_M = hash(d||K_{D1}^+) \tag{2.1}$$

  where, $hash(.)$ represents a secure hash function (e.g., SHA-2 or SHA-3 [118]). Similarly, a hash (H_M^*) will be recovered from the signature by using the following equation:

$$H_M^* = K_M^+(Sig_M) \tag{2.2}$$

  where, K_M^+ is the public key from the OCM. The integrity is verified by comparing H_M with H_M^*.

- Step 2: Once the previous verification is passed, D_i reads the next row j of the RFID content. A hash is now computed on previous stage hash value H_{j-1}, signature Sig_{j-1}, and the public key K_{j+1}^+ using Equation 2.3.

$$H_j = hash(H_{j-1}||Sig_{j-1}||K_{j+1}^+) \tag{2.3}$$

  Similarly, another hash value H_j^* is recovered from the signature by using following equation.

$$H_j^* = K_j^+(Sig_j) \tag{2.4}$$

The verification will pass if $H_j = H_j^*$, j will be increased by 1, and stay at Step 2. Any mismatch of the computed hash value and recovered hash value will give an indication that RFID content is tampered with and the chip will be flagged as recycled, and the corresponding distributor will be identified. In addition, the distributor, D_{i-1} can also be charged for promoting recycled ICs, as either, it does not perform the verification when it acquired the chips from D_{i-2} or deliberately falsified the authentication results. Note that the end-user or the system integrator will also follow the same verification process as D_i.

The authenticity of a device can be ensured by verifying its identity. At every stage (different distributors and the SI), the verification of a device ID is performed. The data d contains an electronic chip ID (See Equation 2.5), which is unique to every device. The manufacturer (considered trusted in our threat model) provides its digital signature to certify $ECID$. Any tampering of d can be detected at any stages (D_1 through SI).

Update RFID Content

In this phase, all entities in the supply chain write data into the RFID memory. As the manufacturer is the first entity in the supply chain and trusted, the content directly written by the OCM should be authentic and verified. On the other hand, the distributors and the system integrator only update the RFID memory, once the chip passes the verification as described above. The manufacturer writes the data d, its public key (K_M^+), and signature (Sig_M) into the RFID tag memory.

The update process for the manufacturers can be described as follows:

- Step 1: The data (d) is constructed by concatenating the RO cycle count (C_0) with measurement conditions (e.g., temperature (T_0), supply voltage (V_0) and duration (t_{D0}), and electronic chip ID ($ECID$).

$$d = \{C_0||T_0||V_0||t_{D0}||ECID\} \tag{2.5}$$

- Step 2: A secure hash is computed based on d and the public key of the first distributor (K_{D1}^+).

$$H_M = hash(d||K_{D1}^+) \tag{2.6}$$

- Step 3: The signature of the manufacturer is computed on H_M.

$$Sig_M = K_M^-(H_M) \tag{2.7}$$

where, K_M^- is the private key of the manufacturer.

- Step 4: Finally, the manufacturer writes the data $\{d, K_M^+, Sig_M\}$ into the RFID and distributes the chip into the supply chain.

The update process for the distributors is described as follows:

- Step 1: D_i reads the entire RFID memory to construct the data (d_i) for hash computation.

$$d_i = \{H(...(H(H(d||K_{D1}^+)||Sig_M||K_{D2}^+)||Sig_{D1}|| \\ ...)||K_{Di}^+)||Sig_{Di-1}\} \tag{2.8}$$

- Step 2: A secure hash is computed on d_i and the public key of the $(i+1)^{th}$ distributor (K_{Di+1}^+).

$$H_{Di} = hash(d_i||K_{Di+1}^+) \tag{2.9}$$

- Step 3: The signature of D_i is computed on H_{Di}.

$$Sig_{Di} = K_{Di}^-(H_{Di}) \tag{2.10}$$

- Step 4: Finally, the distributor appends $\{K_{Di}^+, Sig_{Di}\}$ to the RFID and sends the chips to the next distributor or system integrator.

Finally, the update process for the SI can be described as follows:

- Step 1: SI reads the entire RFID memory to construct the data (d_{SI}) for hash computation.

$$d_{SI} = \{H(...(H(H(d||K_{D1}^+)||Sig_M||K_{D2}^+)||Sig_{D1}|| \\ ...)||K_{SI}^+)||Sig_{Di}\} \tag{2.11}$$

- Step 2: A secure hash is computed on d_{SI}.

$$H_{SI} = hash(d_{SI}) \tag{2.12}$$

- Step 3: The signature of SI is computed on H_{SI}.

$$Sig_{SI} = K_{SI}^{-}(H_{SI}) \tag{2.13}$$

- Step 4: Finally, SI appends $\{K_{SI}^{+}, Sig_{SI}\}$ to the RFID after deploying it into a system.

2.3.2 Approach for Verification of the Prior Usage of an IC

This proposed solution enables a chain of trust among the distributors. Anyone in the supply chain can verify the identity of chips without powering them on. However, the final verification, whether a chip is used before or not, is performed at the system integrator's site. Figure 2.5 shows the verification of the prior usage of an IC by the SI. Once the complete route of an IC in the supply chain is verified, the SI powers up the chip to measure the RO frequency.

The measurement conditions (T_0, V_0) are first set up to measure the RO clock cycle count (C_0^*) with the fixed time interval (t_{D0}). These measurement parameters are reconstructed from the data ($d = \{C_0||T_0||V_0||t_{D0}||ECID\}$), which is present at the 1^{st} row of the RFID memory. The difference between the measured clock cycle count (C_0^*) and the registration clock cycle count (C_0) is calculated. If the difference is greater than the precision of the counter (measurement error, Δ), the chip is identified as recycled, otherwise, as new. The same setup (e.g. a timer and a counter) presented in Figure 2.1 can be used for the measurement, and the details can be found in [1].

2.4 Analysis

2.4.1 Implementation Details

We have implemented our proposed scheme using a commercial off-the-shelf RFID tag (MIFARE Classic [122]). Figure 2.6 shows the experimental setup for implementing our proposed solution. We choose a very low-cost RFID reader (MFRC522 RFID module [123]) to read and

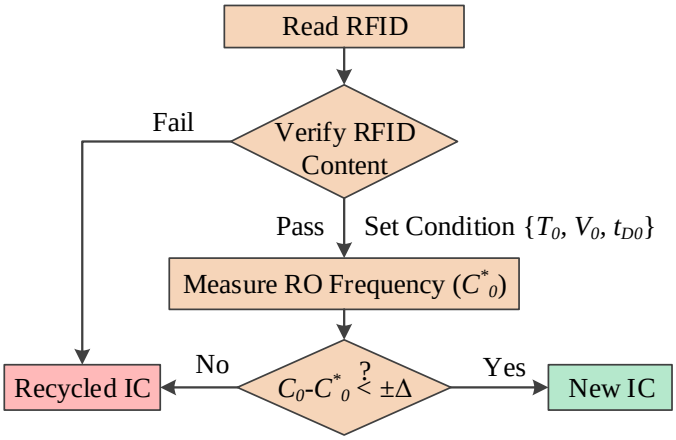

Figure 2.5: Flow for Detecting Recycled ICs.

update the RFID tag memory. A Raspberry Pi 3 with 1.2 GHz quad-core CPU and 1GB RAM, which controls the read/write data, is used to interface the RFID reader. Here the SRAM chip is treated as the circuit under test. We use M2Crypto [124], crypto, and the SSL toolkit for Python, to compute and then verify elliptic curve digital signatures. The distributor IDs and digital signatures are loaded into the MIFARE RFID tag using the MFRC522 RFID module. Note that the Laundry MIFARE Classic RFID tag has an electrically erasable programmable read-only memory (EEPROM) of 1K Bytes, where we store the trace for different distributors. In the MIFARE Laundry Classic 1K RFID tag, there are 16 sectors, and each sector consists of 4 blocks of 16 bytes each.

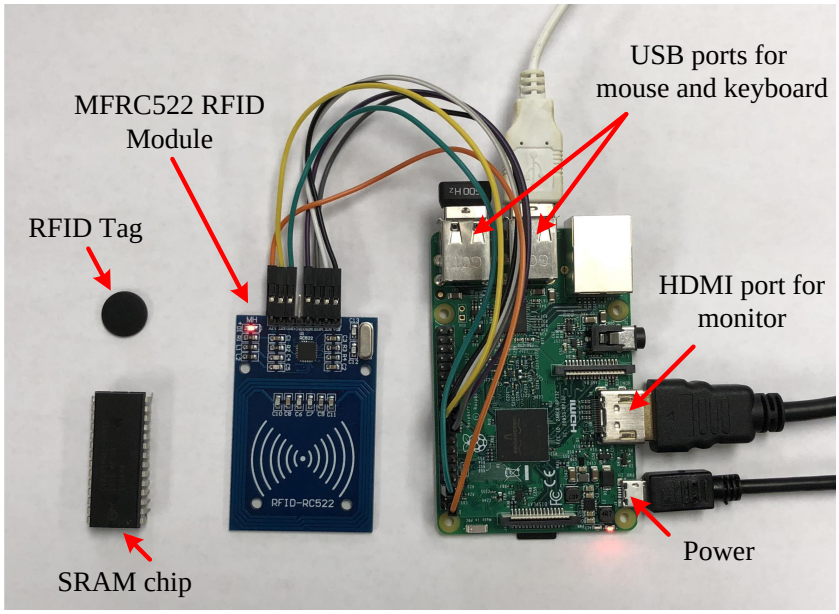

Figure 2.6: Setup for implementing our proposed approach.

The memory space required to store one distributor (and the manufacturer) is determined as follows:

- The data (d) can be of 174 bits (T_0 of 10 bits, V_0 of 10 bits, C_0 of 32 bits, t_{D0} of 10 bits, and $ECID$ of 112 bits [113]).
- A distributor (manufacturer) ID can be of 20 bits, which can represent 2^{20} distinct entities. Note that it is not necessary to store the public keys (K^+) in the RFID due to its resource (low memory) constraints. One can store unique public IDs of the manufacturer and the distributors instead. It is necessary to maintain a website from which one can find the public keys for different manufacturers, distributors, and system integrators using this public ID.
- ECDSA signature (SECP-256R1 ECDSA [125]) can be of 512-bits. We chose SHA256 to hash the data. We used the M2Crypto library [124] in python to create the hash and digital signature. Here the signature is combined with 10 bytes of cyclic redundancy check (CRC) to protect the integrity of the signature from unintended modification, such as, noise.

Combining all these bits, we need 5 blocks in the RFID tag memory to store the trace for one distributor or the system integrator. For the manufacturer, we need to store the data (d) along with its ID and signature, which requires 7 blocks. *We can store relevant information for one manufacturer, one system integrator, and seven distributors when we use this RFID tag.*

Note that we do not need to add one tag per chip, and it is not necessary to attach this tag into the package as long as they travel together in the supply chain. One can add only one RFID tag for a small batch of chips. One only needs to measure all the frequencies and then construct the data ($d = \{C_0^{(1)}||C_0^{(2)}||...||C_0^{(n)}||V_0||T_0||t_{D0}||ECID^{(1)}||ECID^{(2)}||...||$ $ECID^{(n)}\}$) and then store the signature (Sig_M) into the RFID tag. Here, $C_0^{(i)}$ and $ECID^{(i)}$ are the RO frequency and $ECID$ of the i^{th} chip, respectively.

2.4.2 Security Analysis

In this section, we present different attack scenarios against our proposed solution and assess the resistivity of the proposed architecture under such attacks.

Tampering with the RFID Content

In this attack, an adversary tampers with the RFID content using an RFID reader. To break the traceability of a component in the supply chain, the attacker can remove one or more entries from the RFID memory to eliminate the trace for a few distributors. For example, an adversary (D_4) removes the 4^{th} entry (i.e., public key and signature from the D_3) of the RFID tag memory, and then sell the chip to D_5. Figure 2.7 shows an example of removing an attack, where the row highlighted in red was removed by the distributor D_4. Note that M and D_i represents the manufacturer, and i^{th} distributor, respectively.

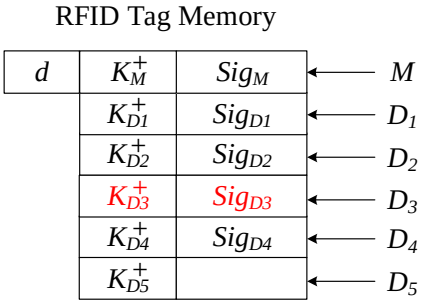

Figure 2.7: Tampering with the RFID content to modify trace.

This attack can be detected by a distributor (in this example, distributor D_5) or the system integrator (SI) while they perform the signature verification. When doing authentication described in Section 2.3, the first two verifications for the manufacturer (M) and distributor (D_1) will be passed. However, the third verification will fail as there will be a mismatch of the computed hash value H_2 and recovered value H_2^* from the signature because of the involvement with different public keys. The authentication can be performed as follows:

- D_5 reads the entire memory, constructs data for each stages, and then compute the hashes.

  $d_M = \{d\}, H_M = hash(d_M||K_{D_1}^+)$

  $d_{D1} = \{H_M||Sig_M\}, H_{D1} = hash(d_{D1}||K_{D2}^+)$

  $d_{D2} = \{H_{D1}||Sig_{D1}\}, H_{D2} = hash(d_{D2}||K_{D4}^+)$

- It recovers the hashes from the signatures.

  $H_M^* = K_M^+(Sig_M); H_{D1}^* = K_{D1}^+(Sig_{D1})$

$H^*_{D2} = K^+_{D2}(Sig_{D2})$

- Finally, it performs signature verification.

  $ver(H_M, H^*_M) = pass;\ ver(H_{D1}, H^*_{D1}) = pass$

  $ver(H_{D2}, H^*_{D2}) = fail$, as

  $H^*_{D2} = hash(H_{D1}||Sig_{D1}||K^+_{D3})$ where the $ver()$ function can be described as follows:

$$ver(x_1, x_2) = \begin{cases} pass & \text{if } x_1 = x_2; \\ fail & \text{otherwise;} \end{cases} \tag{2.14}$$

Impersonation of a Distributor

In this attack, an untrusted distributor (D_j) tries to sneak into stage $(i+1)^{th}$ distribution stage using the identity of a trusted distributor (D_{i+1}). However, this attack is infeasible as the entries of the RFID memory is protected by the digital signature. It is infeasible for D_j to create a signature of D_{i+1}. As a result, D_j cannot pass the chip to D_{i+2} as the signature verification will fail. In addition, the motivation for D_j to sneak into the supply chain without tampering the chip is not clear. Furthermore, it can perform tampering with an authentic chip received from D_i and send to D_{i+1}, which is beyond the scope of this dissertation.

Dictionary Attack

In this attack scenario, a recycler (untrusted distributor) constructs a dictionary of RO frequencies from many new chips. Each entry of the dictionary consists of the data (d), the manufacturer's public key (K^+_M), and its signature (Sig_M) from new chips. After recycling an old chip, the adversary measures the frequency of that RO. If a match (or close enough) is found in the dictionary, he/she can update the RFID content with the respective content from the dictionary. Note that the RO frequencies of the new chips vary significantly (generally Gaussian in nature [67]) due to process variation. It can be possible that two RO frequencies of new and recycled chips are of the same value. Thus,

it seems that a recycler can impersonate an old chip with a new one. However, one can easily detect this attack by verifying the signature (Sig_M). The verification process can be performed as follows:

- Read $ECID^*$ value, and RFID contents from the chip.
- *First Authentication:* This fails if $ECID$, which is present in the data (d) from the RFID, does not match with $ECID^*$ of the chip. This authentication can only be performed by the SI as it is necessary to power up the chip to read $ECID^*$.
- *Second Authentication:* If the *First Authentication* passes, a second authentication is necessary to verify the signature, which can be done as follows: (i) compute hash on data (d), $H_d = hash(d)$, (ii) recover the hash from the signature (Sig_M), $H_d^* = K_M^+(Sig_M)$, and (iii) verify both these hashes using $ver(H_d, H_d^*)$ function (see Equation 2.14).

  Note that this second authentication can be performed by any entity in the supply chain.

Tampering the Ring Oscillator

For this attack scenario, an attacker tampers the physical structure of the RO of a counterfeit chip. An RO becomes faster if the number of inverter stages becomes smaller. An attacker can reduce the number of inverter stages using an FIB circuit edit [126]. To perform this attack, the chip needs to be decapsulated to remove the old package and then perform the edit. After the modification, the chip needs to be repackaged again and remarked to its original specification. Note that this attack needs to be performed on every chip. As a result, the circuit edit, repackaging and remarking may not be cost-effective to the counterfeiters. Hence, it is unlikely to be used in practice by an adversary.

Improper Registration

In this attack scenario, an untrusted entity at the production site can update the RFID content with a false oscillation count which is significantly less than the actual measured value. As a result, the oscillation frequency can still be found very close to the registration value, even though a chip has been used in the field for a long time. However, there will not be any financial motivation behind such an act from a foundry's perspective as it will only help the counterfeiters. Moreover,

we generally consider the foundry as trusted for IC recycling. Thus, manipulating the frequency value in the registration phase does not make any financial motivation to the foundries.

Key Breach

If a breach happens for distributor D_j, it is required to update its keys and put its new signature in forthcoming chips. However, the public key remains unchanged (old key) in the RFID memory of chips with previous signatures. Practically an adversary can put a signature at the $(j+1)^{th}$ location of the RFID memory (the first location is reserved for the manufacturer) by modifying the next stage distributor ID, and thus, make the authentication fail for an authentic chip. At this point, the system integrator (SI) can contact distributor D_j for more information regarding the key breach. It is then up to the SI to decide the acceptance of this chip. If a key breach happens [127–130], the distributor must report it to all the participating entities in the supply chain. Note that if the manufacturer's database is breached, no authentication can be performed for chips with old keys as the RO frequency value can be updated in the RFID memory.

2.5 Summary

In this chapter, we proposed a robust and low-cost solution for detecting recycled ICs, which are reclaimed from old electronic systems. Our solution enables traceability by ensuring a chain of trust among the manufacturer, the distributors, and the system integrator. It utilizes a small passive RFID tag, which needs to be placed on the package of a chip. Any entity in the supply chain can verify the authenticity of a chip using a hand-held commercial RFID reader. It is not necessary for a distributor to power up a chip during the verification process. However, the final verification needs to be performed at the system integrator's site and requires powering up the chip for RO frequency measurement. As an RFID tag can be placed in the package of a chip, our proposed solution can practically be applied to all the chips.

Chapter 3

A Novel Topology-Guided Attack and Its Countermeasure Towards Secure Logic Locking

3.1 Introduction

The prohibitive cost of building and maintaining a foundry (fab) with advanced technology nodes has forced many design companies to become fabless and adopt the horizontal semiconductor integration model. Currently, the majority of the design houses integrate intellectual properties (IPs) obtained from different third-party IP (3PIP) vendors along with its design and outsource the manufacturing to an offshore foundry resulting in a global supply chain with distributed vendors carrying out design, verification, fabrication, testing, and distribution of chips. The involvement of untrusted entities at various stages in the IC manufacturing and testing process has resulted in evident security threats, such as piracy or theft of IPs, overproduction of ICs, and sale of out-of-specification/rejected ICs [2–4, 10, 13, 20, 131, 132]. Many design-for-trust techniques have been studied over the years as countermeasures against the aforementioned threats [7, 8, 10, 19, 20, 133–137].

Among abovementioned countermeasures, logic locking is the most widely accepted and studied design-for-trust technique to prevent threats from untrusted manufacturing and testing. Logic locking hides the circuit's inner details by incorporating key gates in the original circuit resulting in a key-dependent locked counterpart. The resultant locked circuit functions correctly once the secret key is programmed in its tamper-proof memory. Otherwise, it will produce erroneous outputs for the same input patterns, which makes it practically unusable. Over the years, different locking techniques are proposed, which can be primarily categorised based on key-insertion strategy (see Figure 3.1) and can be described as – (i) XOR-based, (ii) MUX-based, (iii) Look-up Table (LUT)-based, and (iv) state-space based. However, XOR-based logic locking is popular due to its simplicity.

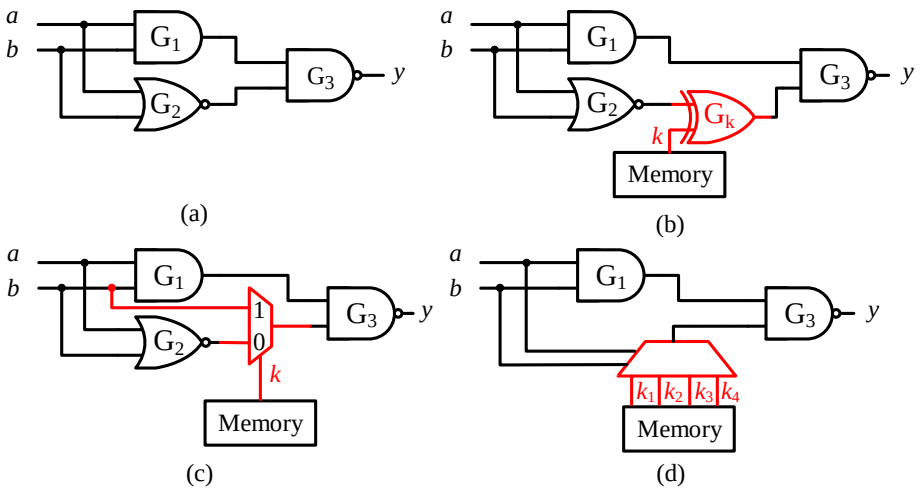

Figure 3.1: Logic locking methods: (a) An original netlist (b) XOR/XNOR-based logic locking (c) MUX-based logic locking (d) LUT-based logic locking.

The research community is continuously driven to reveal logic locking vulnerabilities through attacks and to propose countermeasures in turn. The majority of early work was demonstrated vulnerable by oracle-guided key-pruning attacks [138] and its variants [32, 33, 139, 140]. Since then, many SAT-resilient solutions have been proposed [20–22, 141–147]. However, some of them have been broken as well [40, 85, 139, 148–152]. A hardware Trojan insertion-based attack can defeat any logic locking techniques that rely on storing the secret key in a tamper-proof memory [18, 43]. Even though the SAT attacks are widely popular amongst the research community, the attack model assumes the availability of an oracle or a functionality correct (unlocked) IC pre-loaded with the correct key, and the adversary has the scan-chain access to obtain the input-output responses. This serves as the limitation as many of the chips used in critical or DoD applications are highly unlikely to be circulated (unless it is a commercial-off-the-shelf, COTS part) in the market right after manufacturing. In addition, the concept of restricting scan access (which will limit the access of input/output pairs of an oracle by the attacker) has also been adopted to provide security against SAT attacks. An adversary is not restricted to performing only SAT-based attacks as it may deploy other effective attacks to extract the secret key from a locked netlist. Therefore, it is necessary to consider and explore the different directions by which an untrusted foundry can exploit security vulnerabilities to undermine the security of logic locking.

In this dissertation, we propose a novel oracle-less attack on logic-locked circuits to determine the key. Exploring the capabilities of an adversary, ***is it possible to determine the secret key simply by analyzing the circuit topology?*** The answer is yes, as the entire circuit topology is built from basic Boolean functions that are repeated multiple times. An adversary can determine the secret key by comparing the locked instances of these functions with the unlocked instances in the entire netlist. This proposed attack is an oracle-less self-referencing attack. Our proposed attack is denoted as ***TGA***: ***T***opology-***G***uided ***A***ttack on logic locked circuits. By using our proposed attack, the secret key can be estimated efficiently even for the circuits that the SAT attack fails (see in Section 3.5 for *c6288* circuit). In addition, an adversary can unlock any netlist using our proposed attack without waiting for a working chip available in the market or with no scan access. This was further validated and demonstrated at *UF/FICS Hardware De-obfuscation competition at Trusted and Assured Microelectronics (TAME) forum [153, 154].*

3.2 XOR-based Logic Locking

To describe our proposed topology-guided attack based on function search, it is necessary to present XOR-based logic locking. Additionally, one must analyze the resulting circuit modifications based on the selected correct key bit and the key gate type (either XOR or XNOR) to lock the original functionality. This will assist in building equivalent unit functions (*EUFs*) that will be searched in the netlist to perform the proposed attack.

Figure 3.2 shows an example to lock a circuit using an XOR gate, which has three inputs (X_1, X_2 and X_3) and one output (Y). One key gate with value k is selected to obfuscate the functionality of the circuit. The original circuit is shown in Figure 3.2.(a). There can be two possible key values, $k = 0$ and $k = 1$. For $k = 0$, an XOR gate can directly be placed at node X_4, which is shown in Figure 3.2.(b). However, for $k = 1$, two possible scenarios may occur. One can invert the previous stage functionality, which is shown in Figure 3.2.(c). It is also possible to modify successive stage function using DeMorgan's Theorem, shown in Figure 3.2.(d).

In this example, the original function of the circuit is $Y = X_3 \cdot X_4$, where $X_4 = X_1 \cdot X_2$. It is not necessary to change the functionality of the preceding or succeeding stages of the XOR gate when k = 0.

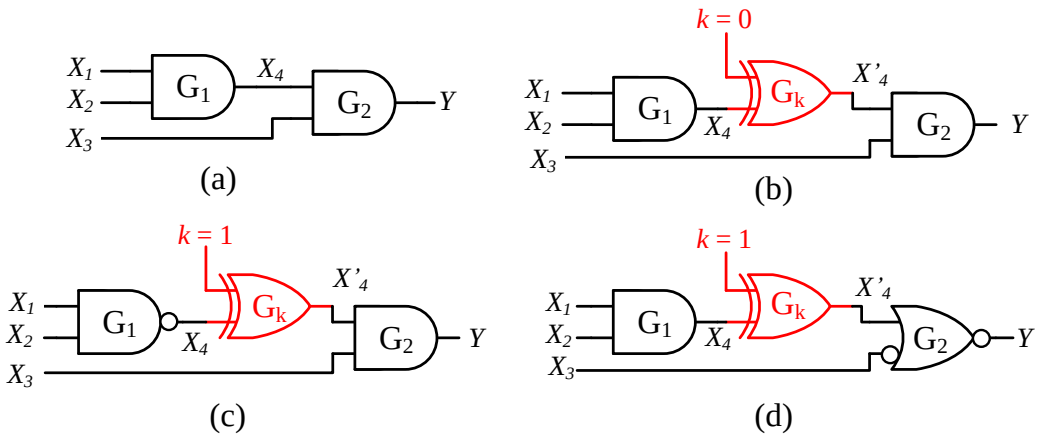

Figure 3.2: Logic locking using Exclusive OR (XOR) gates. (a) Original netlist. (b) Locked netlist when k = 0. (c) *Case-I*: Locked netlist when k = 1. (d) *Case-II*: Locked netlist when k = 1 (using DeMorgan's Theorem).

$$X_4^{'} = X_4 \oplus 0 = X_4 = X_1 \cdot X_2 \tag{3.1}$$

To preserve the original functionality for k = 1, it is required either to invert the functionality of the preceding stage (Figure 3.2.(c)) or compensate the functionality of the following stage (Figure 3.2.(d)) of the added XOR gate. For the first case, the original functionality preserves as $X_4^{'} = 1 \oplus \overline{X_4} = X_4$. For the second case, DeMorgan's transformation is necessary as shown below:

$$Y = \overline{\overline{X_3} + X_4^{'}} = \overline{\overline{X_3}} \cdot \overline{X_4^{'}} = X_3 \cdot \overline{(1 \oplus X_4)} = X_3 \cdot X_4 \tag{3.2}$$

Note that only XOR gates are used in the example to lock the netlist. However, one can also use XNOR gates for such purposes, which have the opposite logic function compared to the XOR gate. It is important to remember that one cannot insert an XOR gate with $k = 0$ or XNOR gate with $k = 1$ for every key bit, as the adversary can determine the secret key just by observing the type of key gates.

3.3 Proposed Topology-Guided Attack on Logic Locking

The general locking strategy adopted to provide security in a circuit includes the placement of key gates either randomly or in some particular manner (e.g. pair-wise). Since, the secret key

associated with the key gates is the same for all the chips manufactured with the same design, finding this key from one netlist undermines the security resulting from logic locking. In this section, we show how an adversary can easily extract the secret key for a key-based locked design using our proposed oracle-less and topology-guided attack, which is built on searching the hypothesis key-based equivalent unit function in the entire locked netlist. Moreover, this attack overcomes the limitations of SAT attacks that require an oracle with scan access. This dissertation will present the different steps involved in performing the proposed attack.

3.3.1 Adversarial Model

The specific objective is to undermine the security of a logic locking technique by determining the secret key. The secret key is stored in a secure and tamper-proof memory so that the adversary cannot access the key values directly from an unlocked chip. The adversarial model is presented to clearly state the resources and the assets possessed by an adversary. In our attack model, the adversary is assumed to be an untrusted foundry and has access to the following:

- *Gate-level netlist:* As the primary attacker, the foundry can have access to the gate-level netlist of a locked IC. The SoC designers typically send the circuit layout information using GDSII or OASIS files [155] to a foundry for chip fabrication. With the help of advanced tools, the foundry can extract the gate-level netlist from those provided GDSII/OASIS files [156].
- *Location of the key gates:* The location of key gates can be determined as these gates are connected either directly or through temporary storage elements to the tamper-proof memory. An adversary can easily track the routing path from the tamper-proof memory to the corresponding gates to determine their locations.
- *Locked unit function:* It is trivial for an untrusted foundry to construct equivalent unit functions *EUFs* for launching the topology-guided attack, as it has the netlist and locations of the key gates.

3.3.2 Construction of Equivalent Unit Function

Our proposed attack constructs an equivalent unit function to perform the search. While constructing the *EUF*, an adversary may encounter two different cases, either there is only one key gate, or there are multiple key gates in the *UF*. In either case, the (*EUF*) is constructed using

one/more hypothesis key bits or a combination of hypothesis key bits, and searches that *EUF* in the entire netlist to find a match. The hypothesis key bits will be the correct secret key bits for the respective *UF* if a match is found corresponding to the *EUF*. Otherwise, it constructs another *EUF* using a different combination of values for the hypothesis key bits in both the cases and searches the netlist again. The number of *EUFs* depends on the number of key gates included in the *UF*. In this section, we show how *EUFs* are created to determine the secret key for both *RLL* and *SLL* circuits.

Random Logic Locking

In random logic locking (*RLL*), the key gates are inserted randomly inside the circuit that needs to be protected. In a large design with thousands of gates, it is highly unlikely that multiple key gates will be inserted adjacent to each other. Thus, the inserted key gates usually can be considered individually to construct the equivalent unit functions.

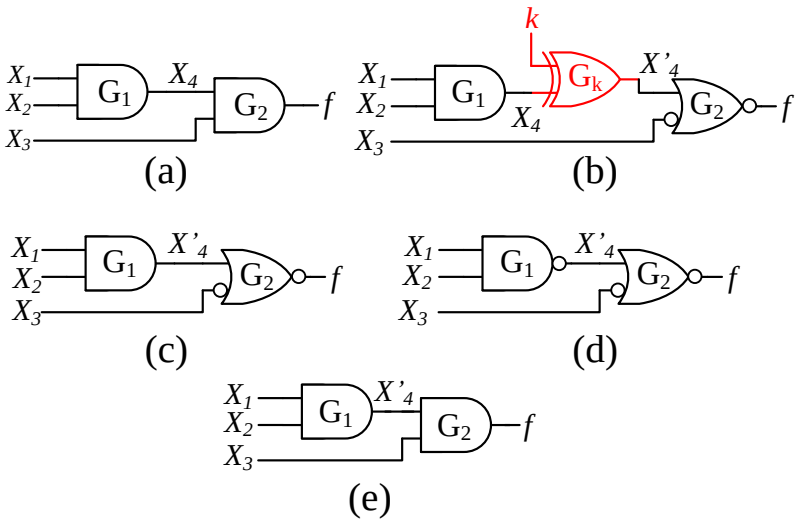

Figure 3.3: *EUF* construction for different hypothesis key values. (a) Original unlocked netlist. (b) Netlist is secured with key value $k = 1$. (c) EUF_0 for hypothesis key $k_h = 0$. (d) EUF_1 for hypothesis key $k_h = 1$ (*Case-I*). (e) $EUF_{\hat{1}}$ for hypothesis key $k_h = 1$ (*Case-II*).

Figure 3.3 illustrates the construction of equivalent unit functions with a single key gate, which can be used to launch the function search attack. Figure 3.3.(a) represents an original unit function to be locked using a correct secret key $k = 1$. The locked circuit is shown in Figure 3.3.(b). The adversary cannot deduce the value of the key, simply by observing the key gate. He/She first makes an assumption for $k_h = 0$, and constructs the *EUF*, which is shown in Figure 3.3.(c). He/She then

searches this function in the locked circuit to find a match. If no match is found (as the actual key is 1), the adversary will construct another *EUF* for $k_h = 1$. Two possible scenarios may occur. For *Case-I*, the output of the previous stage needs to be inverted (shown in Figure 3.3.(d)). On the other hand, a DeMorgan's transformation needs to be carried out to obtain the *EUF* for $k_h = 1$ for *Case-II*, which is shown in Figure 3.3.(e). As inferred from the construction of the equivalent unit function, each key gate has two hypothesis keys with three transformations represented as: (i) EUF_0 where the hypothesis key $k_h = 0$, (ii) EUF_1 (*Case-I*) where the hypothesis key $k_h = 1$, and (iii) $EUF_{\hat{1}}$ (*Case-II*) where the hypothesis key $k_h = 1$ but the modification is carried out using DeMorgan's Theorem. Three *EUFs* can be constructed for a single key bit. For a hypothesis key bit $k_h = 0$, a single implementation of *EUF* can be considered, whereas, two different implementations can be possible for hypothesis key bit $k_h = 1$. As a result, for an *UF* locked with a j-bit key, 3^j number of implementations can be possible. We need to perform all 3^j *EUF* searches to determine the j-bit key.

Strong Logic Locking

The objective of strong logic locking (*SLL*) is to maximize the interference between different key gates to restrict key sensitization at the output [77]. In *SLL*, two or more key gates are inserted adjacent to each other so that their outputs converge at the next stage logic gate. The propagation of one of the key-bit will be possible only if certain conditions are forced on other key inputs or they are known. As these key inputs are not accessible by the attackers, they cannot force the logic values necessary to sensitize a key. As a result, the proposed *TGA* on *SLL* requires an equivalent unit function search with multiple keys instead of a single one for random logic locking.

Figure 3.4 illustrates the construction of *EUFs* with multiple key gates that will assist in performing the function search. The original unit function (as shown in Figures 3.4.(a)) is locked with three key gates to increase the inter key-dependency. The locked unit function is shown in Figure 3.4.(b) with correct key $k_1k_2k_3 = 101$. As an adversary cannot extract the correct key value from the non-volatile memory directly, all the *EUFs* will be constructed and searched in the entire locked netlist. However, the number of constructed *EUFs* will increase due to the number of key gates and its combination in the locked *UF*. As mentioned earlier, each key gate results 3 different *EUFs* (e.g., EUF_0, EUF_1, and $EUF_{\hat{1}}$ based on the hypothesis key values (either 0

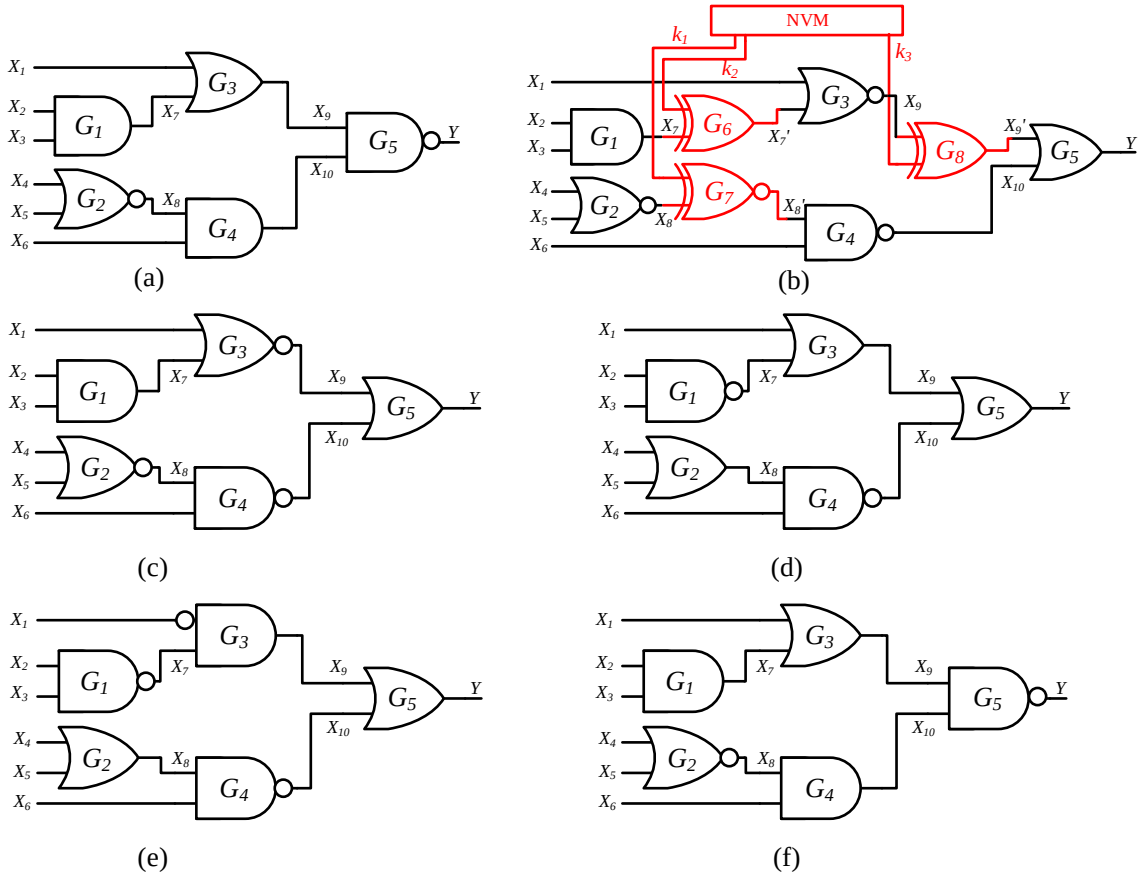

Figure 3.4: Equivalent unit functions for multiple gates with different hypothesis keys. (a) Original netlist. (b) Locked netlist with key value $k_1k_2k_3 = 101$. (c) EUF_{100} for hypothesis key, $k_h = 100$. (d) EUF_{011} for $k_h = 011$. (e) $EUF_{0\hat{1}0}$ for $k_h = 010$. (f) $EUF_{10\hat{1}}$ for $k_h = 101$.

or 1). This will result in overall 27 *EUFs* (i.e. 3^3, as $j = 3$ is the number of key gates in the *UF*) for Figure 3.4.(b), amongst which only four of them are shown in Figure 3.4.(c)-(f). These *EUFs* are derived from different key combinations. For example, EUF_{100} in Figure 3.4.(c) is constructed with hypothesis key bits-based transformation as EUF_1, EUF_0 and EUF_0 for key gates G_7, G_6 and G_8 respectively. Also, EUF_{011} is constructed as shown in Figure 3.4.(d). if it is transformed based on the hypothesis key bits $k_h = 011$ for key gates G_7 (EUF_0), G_6 (EUF_1) and G_8 (EUF_1). Figure 3.4.(e) shows yet another *EUF* represented as $EUF_{0\hat{1}0}$, where the hypothesis key is 0, 1 (Case-II) and 0. Likewise, if we select the transformation as EUF_1, EUF_0 and $EUF_{\hat{1}}$ (Case-II) for key gates G_7, G_6 and G_8 respectively, then we will get the $EUF_{10\hat{1}}$ shown in Figure 3.4.(f). Once all the *EUFs* are constructed, all of them will be searched in the netlist to find a match. As Figure 3.4.(f) is identical to Figure 3.4.(a), the hypothesis key combination $k_h = 101$ should be

the correct key value. If no such match is found for any of the *EUFs*, an adversary cannot make the prediction on the key combination resulting the *UF* being unique in the circuit.

3.3.3 Function Search Using DFS Algorithm

An efficient search algorithm has been developed to search for the *EUFs* in the locked netlist. The structure of a circuit can be transformed and represented as a directed graph, and all the algorithms that can be used to search for the components in a directed graphs can also be applied to search the *EUF*. Therefore, we propose to use the Depth-First-Search (*DFS*)-based algorithm to launch the attack. Generally, the *DFS* method follows the rule: in the graph traverse procedure, the edge from the most recently reached and connected vertex that still has unexplored edges will always be selected as the next edge [111]. Before performing the *DFS*-based search, a data object structure needs to be defined to store and transform the netlist as a directed graph. The gate object needs to have the following attributes: gate type (e.g. XOR, AND, etc.), name of the gate (i.e., its identification in the netlist), an array that contains its preceding gates (i.e. its inputs), and an array that contains its following gates (i.e. its outputs). Then the circuit structure can be transformed and stored into a dictionary, in which the keys are the types of the gates and the values are corresponding gate objects. This dictionary is basically a data structure that stores mappings and relationships of data [157]. The use of a dictionary makes the search for a specific type of gate more efficient.

The procedure of *DFS*-based search is described in Algorithm 1. The function *FS* finds matches for any function (*Fn*) in a circuit that it is taken as an input. Whenever a specific *Fn* needs to be searched in this netlist, we define the last gate of the *Fn* as the root gate (Line 2 in the Algorithm 1). An example root gate is G_2 in the Figure 3.3). All the gates that have the same type as the root gate (G_2) in the dictionary (Line 3) are stored in an array. The *DFS* is then performed on all these found gates (Line 3-7). Finally, all the *UFs* in the netlist will be found and the count of the *F* will be returned as the output (Line 8). The detailed implementation of the *DFS* is demonstrated in Lines 9-38.

The algorithm is implemented with Python 2.7 [158]. The worst-case time complexity of the search algorithm is $O(n * u)$, where n is the size of the netlist and u is the size of a unit function. This is an acceptable complexity since it is known that the subgraph isomorphism problem is an

NP-complete problem and its time complexity is quadratic in the number of nodes [155, 159]. Note that the optimization of the algorithm complexity is not the major objective of this dissertation. However, our search strategy slightly reduces the search complexity by using a dictionary to locate root gates. In this case, the algorithm performs similar to a subtree isomorphism search (or a sequence of tree isomorphism searches), whose complexity is known to be at least subquadratic [160]. Reading the netlist and transforming it into a dictionary may have different complexity. The complexity analysis does not consider the complexity of constructing a netlist dictionary.

3.3.4 Proposed Attack Using Equivalent Unit Function Search

The objective of the proposed topology-guided attack is to recover the entire original netlist using the equivalent unit function search (*FS*). Algorithm 2 describes the proposed attack. The locked circuit (C^*) is given as the input, and the list of predicted key values (K_P) with the success rate (*SR*) will be returned as outputs. K_P contains the predicted value of each key gate, which can be either 0, 1, or X. The X represents an unknown value when the search fails to find a match and make the prediction. The locations of the key gates can be found by tracking the routes originated from the tamper-proof memory, and their numbers can be determined as $|K|$. In order to determine the key value inside a particular *UF*, different *UFs* need to be constructed based on the number of key gates inserted in this *UF*. In addition, each of the key gates comes with a hypothesis key value (either 0 or 1), and this also leads to the different hypothesis key combinations when there are multiple key gates inserted in a *UF*.

Algorithm 1: Function *FS*
Function search based on *DFS* Algorithm.

Input : The gate-level netlist of a circuit (C), Function (Fn)
Output : Result List (L_R)

```
Read C and Fn, and transform them into dictionaries, O and T;
R ← Fn.root; L_S ← O[R.type]; L_R ← φ ;
for each gate G in L_S do
    if DFS(R, G) then
        L_R.append(G);
    end
end
return L_R;
```

```
9  Function DFS(r, g):
10     F ← True;
11     L1 ← r.PrecedingGates; L2 ← g.PrecedingGates;
12     T1 ← L1.types; T2 ← L1.types;
13     if L1 is empty then
14         return True;
15     end
16     for each gate type T in T1 do
17         if gate type T not in T2 then
18             return False;
19         else
20             T2.remove(T)
21     end
22     for each gate RN in L1 do
23         LT ← φ ;
24         for each gate GT in L2 do
25             if GT.type = RN.type then
26                 LT.append(GT);
27             end
28         end
29         FT ← False;
30         for each gate GN in LT do
31             if DFS(RN, GN) then
32                 FT ← True;
33                 break
34             end
35         end
36         F ← F * FT;
37     end
38 return F
```

For each key gate k_i, the *UF* will be constructed based on the value l. Here, l denotes how many layers of gates are considered when constructing the *UFs*. The l is initialized as 1 at the beginning (Line 6), which is also shown in Figure 3.3. Next, the *UF* based on the k_i and l will be generated (Line 7), and the number of key gates (includes k_i) in this unit function will be determined as j (Line 8). The hypothesis key combinations for all the key gates in this unit function will be generated and stored in a list J (Line 9). Note that the order of the keys has no relationship with the real sequence in the circuit, and the number of the combinations is 2^j. Once the key combination list is generated, all the possible *EUFs* will be constructed based on the hypothesis key combinations (Line 11). For each key gate, three different cases need to be considered (see

Figure 3.3 for details), thus 3^j *EUFs* will be generated. The function search (*FS*) (described in section 3.3.3) is then performed to find the repeated instances of *EUFs* (Line 12-14). 2^j count values will be accumulated in a list R (initialized with all 0 in Line 10) for all key assumptions.

Upon finishing the search of all the *EUFs*, if only one count value in R is non-zero, this non-zero value corresponding *EUF* represents a correct key prediction. The hypothesis key J' of this *EUF* will be written into K_P, and the prediction counter (p_c) will be increased by the length of this hypothesis key, j (Line 16-22). Note that, if the key gate is placed in a fan-out net, an additional process needs to be performed (Line 19-21). Function $FV(\ )$ verifies the key decision on each path. It may happen that different paths for the same key gate may have different key predictions. As a result, no prediction will be made in case of any two (or more) paths provide the opposite key-value predictions (Line 36-37). Correct predictions will only be made if different paths make the same prediction (Line 38-39).

On the other hand, if all of the elements in R are equal to 0, this means this unit function is unique in the circuit and the adversary cannot make a prediction on the key value. As a result, an unknown value (X) is assigned to all the j key gates in this *UF*, and the values are also stored in K_P (Line23-25). In the case of multiple count values in R are non-zero, the adversary can neither make the key-value prediction based on the current *EUF*. It is necessary to increase the size of the *EUF* by increasing the layer of gates considered in *EUF* constructions. Therefore, the l value needs to be increased by 1, and the entire searching procedure will be re-performed (26-28).

$$SR = \frac{p_c}{|K|} \times 100\% \tag{3.3}$$

Algorithm 2: Topology-guided attack using *FS*

Input : Locked Circuit Netlist (C^*)
Output : List of predicted key values (K_P), Success Rate (SR)

1 Read the netlist C^*;
2 Determine the location and number $|K|$ of key gates;
3 Initialize correct prediction counter, $p_c \leftarrow 0$;

```
4  for i ← 1 to |K| do
5      if k_i is not determined in K_P then
6          Initialize layer counter, l ← 1;
7          Get the unit function for k_i based on l ;
8          Get number of key gates j in the function;
9          J ← key combinations list, where the length of J = 2^j;
10         R ← [0] * 2^j ;
11         Generate 3^j equivalent unit functions for the j-bit key ;
12         for each generated EUF do
13             J' ← hypothesis key of EUF;
14             R[J.index(J')] ← R[J.index(J')] + FS(C*, EUF).sz();
15         end
16         if R.nonzero = 1 then
17             J' ← R.index(1);
18             Correct hypothesis key k_j ← J[J'] ;
19             if Any key gate in k_j is placed in a fan-out net then
20                 k_j = FV( );
21             end
22             Write k_j into K_P; p_c ← p_c + j;
23         else if R.nonzero = 0 then
24             k_1...k_j ← X;
25             Write k_1...k_j into K_P;
26         else
27             l ← l + 1, go to line 7
28     else
29         Continue
30 end
31 Compute success rate, SR ← (p_c / |K|) × 100%;
32 Output K_P, SR;
33 Function FV( ) :
34     Construct different EUFs for the fanout paths;
35     Search EUFs for each path and make key prediction ;
36     if Opposite predictions for different paths then
37         k_i ←X ;
38     else if Same predictions for different paths then
39         k_i ← {0 or 1};
40     end
41 return k_i
```

Finally, the success rate is computed using Equation 3.3. Here, $|K|$ presents the size of the key while p_c indicates the value stored in the correct prediction counter. The algorithm will finally report the predicted key list K_P and *SR* (Line 32).

The proposed attack may also cause incorrect predictions. For example, it is possible that the actual key bit is 1 when the attack gives an estimation as 0, and vice versa. It is thus necessary to measure the accuracy of the proposed attack. The misprediction rate (*MR*) of our proposed attack can be described as the ratio of the incorrect predictions to the key size and is presented using the following equation:

$$MR = \frac{p_i}{|K|} \times 100\% \tag{3.4}$$

where, p_i denotes the total number of incorrect predictions.

3.4 Countermeasure for TGA

In this section, we propose an effective key insertion algorithm, which can prevent the proposed topology-guided attack. As an adversary performs *EUF* search in the netlist to find out the reference *UF*, this attack can be prevented if the search of those key gates and *EUFs* always returns no results or contradictory values. The basic idea of the countermeasure is to lock all the repeated instances of *UFs* and insert the key gate(s) in all unique *UFs* in the circuit simultaneously. As a result, the adversary cannot predict and recover the correct key values by comparing the locked *UFs* with the unlocked version. In order to find all the repeated instances of selected *UF*, the *UF* search will be performed at the beginning before the key gates are placed into the netlist.

3.5 Simulation Results and Discussions

This section presents the results and evaluate the performance of our proposed topology-guided attack on different logic locking schemes. We provide an in-depth analysis of key prediction accuracy of the proposed attack on ISCAS'85 [108] and ITC'99 [109] benchmark circuits locked with *RLL* and *SLL* using our in-house script. In addition, we have also validated our proposed attack on TrustHub benchmark circuits [110].

Algorithm 3: Insertion of key gates to prevent topology-guided attack

Input : Gate level netlist of a circuit (C),
Key size ($\langle K_{min}, K_{max} \rangle$)

Output : Locked netlist (C^*) and Key value (K^*)

Initialization: $n \leftarrow 0$, $r \leftarrow 0$;
while $n < K_{min}$ **do**
  Select a root gate randomly from C;
  Construct the unit function, UF;
  $r \leftarrow FS(C, UF).sz()$;
  if *RLL* **then**
    if $r = 1$ **then**
      Insert the
       key gate at one random input of root gate and assign key value, $k_n \in \{0, 1\}$;
      Write key value, $K^*[n] \leftarrow k_n$;
      $n \leftarrow n + 1$;
    else if $1 < r \leq K_{max} - n$ **then**
      Lock all the *UFs*;
      Write key values to $K^*[n + r : n]$;
      $n \leftarrow n + r$;
    end
  else if *SLL* **then**
    if $r = 1$ **then**
      Insert j
       key bits in the unique UF, and assign key values, $k_n,\ k_{(n+1)},\ \ldots,\ k_{(n+j)}$;
      Write key value, $K^*[n + j : n] \leftarrow [k_{(n+j)}, \ldots,\ k_{(n+1)}, k_n]$;
      $n \leftarrow n + j$;
    else if $1 < r \leq K_{max} - n$ **then**
      Lock all the *UFs*;
      Write key values to $K^*[(n + r * j) : n]$;
      $n \leftarrow n + r * j$;
    end
  end
end
Output C^* and K^*;

3.5.1 Performance Analysis

Four different benchmark circuits, *c6288*, *c5315*, *b15*, *b17* were first selected for determining the success rate (*SR*) and misprediction rate (*MR*) of our proposed *TGA*. We have created 100 instances of the locked circuit based on *RLL* and *SLL*) for each benchmark circuits, where 128 key gates were

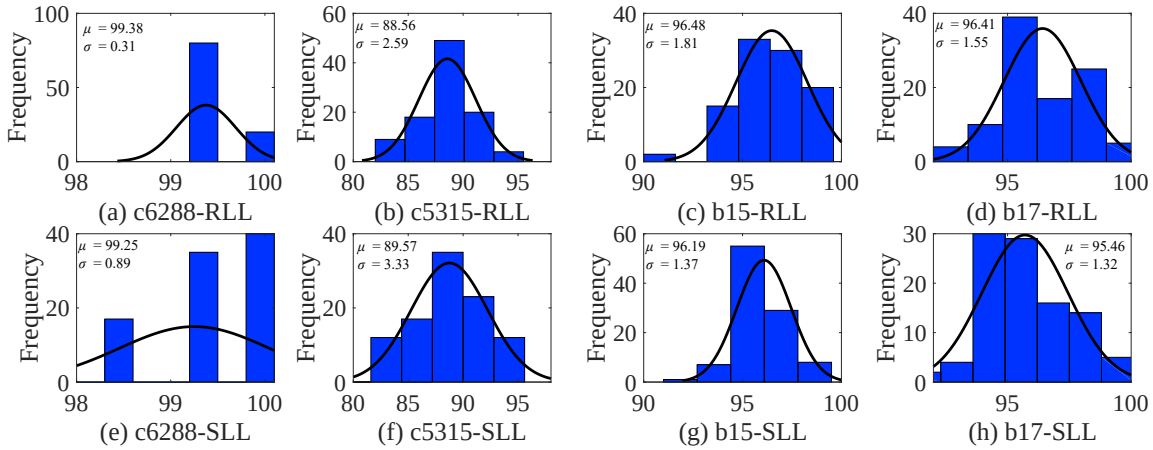

Figure 3.5: Histogram plots of the *SR* for different benchmark circuits with 128 key bits: (a) *c6288-RLL* (b) *c5315-RLL* (c) *b15-RLL* (d) *b17-RLL* (e) *c6288-SLL* (f) *c5315-SLL* (g) *b15-SLL* (h) *b17-SLL*.

placed, and then attacked using Algorithm 2. For each locked circuit, the success rate (*SR*) is computed using Equation 3.3, while the misprediction rate *MR* is calculated using Equation 3.4. In general, the mean and standard deviation are presented by μ and σ for Gaussian distributions related to *SR*, whereas they are all represented by λ^{-1} for exponential distributions that is related to *MR* plots.

Figure 3.5 shows the histogram plots of *SR* metric for the four selected benchmark circuits based on *RLL* (see Figure 3.5.(a)-(d)) and *SLL* (shown in Figure 3.5.(e)-(h)). For benchmark circuit *c6288-RLL*, we estimate the majority of the key bits (Figure 3.5.(a)) as this multiplier consists of many half and full adders. 127 out of 128 key bits were predicted successfully, which resulted in a minimum *SR* of 99.22%. Figure 3.5.(b) shows the *SR* distribution for the *c5315-RLL* circuit. A Gaussian distribution is observed with μ of 88.56% and σ of 2.59. Similar behavior was observed for the other two benchmarks circuits as shown in Figure 3.5.(c) and Figure 3.5.(d). The μ for $b15-RLL$ and $b17-RLL$ are 96.48% and 96.41% with the σ values as 1.81 and 1.55 respectively. We observe a similar Gaussian distributions for the *SR* on locked circuits using SLL (see Figure 3.5.(e)-(h)).

The histogram plots of misprediction (*MR*) for the same selected benchmark circuits are presented in Figure 3.6. Figures 3.6.(a)–(d) present the *MR* plot for the circuits locked with *RLL*. For *c6288-RLL* benchmark circuit, all the key bits can be determined correctly with a 0% *MR* in majority of the cases. The worst case is one bit misprediction, resulting in maximum value of *MR* within 1%. As for *c5315-RLL*, we observe an exponential distribution with a mean and standard

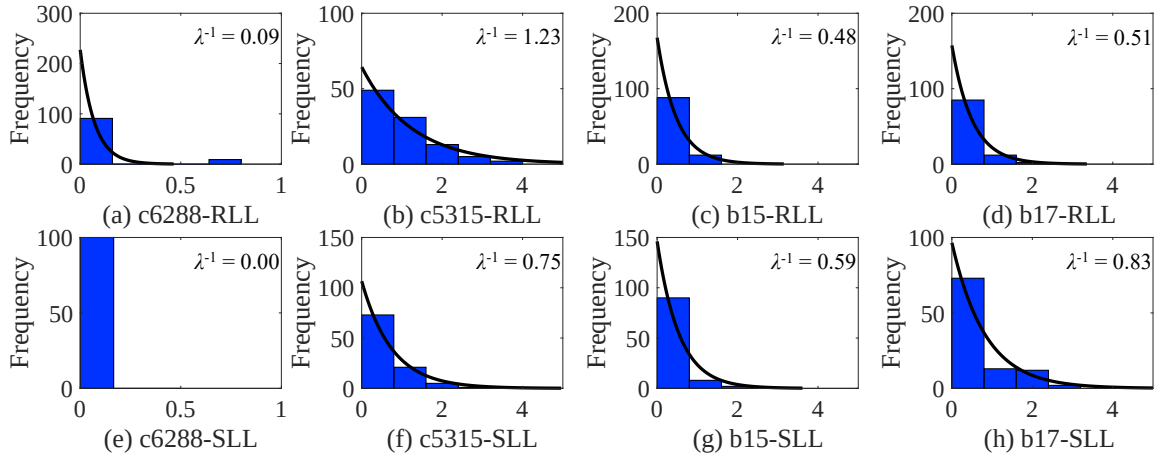

Figure 3.6: Histogram plots of the *MR* for different *RLL* and *SLL* benchmark circuits with 128 key bits: (a) *c6288-RLL* (b) *c5315-RLL* (c) *b15-RLL* (d) *b17-RLL* (e) *c6288-SLL* (f) *c5315-SLL* (g) *b15-SLL* (h) *b17-SLL*

deviation (λ^{-1}) of 1.23. As observed from Figure 3.6.(c) and Figure 3.6, *b15-RLL* shows λ^{-1} of 0.48%, whereas *b17-RLL* shows λ^{-1} of 0.51. Likewise, a similar analysis can be done for *MR* for the same selected benchmark circuits locked with *SLL* plotted in Figure 3.6.(e)–(h).

Table 3.1: Success rate (*SR*) and misprediction rate (*MR*) for estimating keys for RLL and SLL circuits.

Benchmark	**# Total Gates**	**# Key Gates**	**RLL**			**SLL**		
			SR (%)		**MR (%)**	**SR (%)**		**(MR) (%)**
			μ	σ	λ^{-1}	μ	σ	λ^{-1}
c880	404	32	75.03	6.14	1.53	74.63	5.89	1.59
c1350	593	32	69.25	5.97	0.00	69.10	6.41	0.00
c1908	768	32	74.13	4.08	1.44	73.66	4.53	1.72
c2670	1193	32	75.22	4.98	1.13	74.78	5.07	1.06
c3540	1669	128	80.39	3.72	1.76	80.56	4.63	2.01
c5315	2307	128	88.56	2.59	1.23	89.57	3.33	0.75
c6288	2406	128	99.38	0.31	0.09	99.25	0.89	0.00
c7552	3512	128	91.08	3.17	2.03	90.21	2.79	0.97
b14	3461	128	94.16	2.00	0.52	93.67	2.10	0.92
b15	6931	128	96.48	1.81	0.48	96.19	1.37	0.59
b20	7741	128	97.17	1.44	0.25	96.95	1.48	0.84
b21	7931	128	95.40	1.71	0.35	94.50	1.86	0.63
b22	12128	128	96.34	1.25	0.37	95.78	1.62	0.77
b17	21191	128	96.41	1.55	0.51	95.46	1.32	0.83
b18	49293	128	90.25	2.54	0.29	89.36	2.96	0.80
b19	98726	128	89.56	3.06	0.45	88.11	3.55	0.95

Table 3.1 shows the success rate (*SR*) and misprediction rate (*MR*) of our proposed attack on different ISCAS'85 [108] and ITC'99 [109] benchmark circuits locked with *RLL* and *SLL* techniques. The number of logic gates in the circuit and inserted key gates are presented in Columns 2 and 3, respectively. The number of key gates is set to 32 for the first four benchmark circuits (e.g. c880, c1350, c1908, and c2670), while the remaining are inserted with 128 key gates. Columns 4 and 5 presents the mean value μ and standard deviation σ of *SR* values (see Equation 3.3) by analyzing 100 locked instances for each benchmark circuit to determine the accuracy of the proposed *TGA* (see Algorithm 2 for details) for *RLL*. For *c5315* benchmark, 128 key gates are inserted randomly in the netlist with 2307 logic gates. The μ of success rate *SR* is 88.56%, and the σ is 2.59%, which means that the confidence of 99.7% (for $\pm 3\sigma$) can be observed in the range from 80.79% to 96.33%. A similar analysis can be performed for all the benchmarks shown in each row. For the larger benchmark circuits, the average success rate *SR* can be increased by over 90% because of the increased search space, which makes our proposed *TGA* efficient for larger designs. Note that, although *SAT* fails on benchmark c6288, our proposed attack provides better accuracy (average of 99.38%) for benchmark *c6288* due to its special topology – it is a multiplier, which consists of 225 full adders and 15 half adders. Therefore, an adversary can choose our proposed attack as an alternative to the SAT attacks.

Column 6 shows the mean (or standards deviation) *MR* of each benchmark circuit, calculated using Equation 3.4. Note that the mean and standards deviation are of the same value for an exponential distribution. The average *MR* is less than 1% for most benchmark circuits, which makes our attack very useful for determining the secret key. Table 3.1 also presents the *SR* and *MR* metrics for the same benchmark circuits locked with *SLL*, and shown in Columns 7 to 9. We also observe a similar trend like *RLL*.

In order to evaluate the accuracy of *SR* and *MR* with the circuit size, scatter plots of *SR* and *MR* are performed and shown in Figure 3.7. A least-squares trend line is added in every plot, while m represents the line's slope. In Figure 3.7(a), we observe a slope m of 9.60, which indicates that the value of *SR* increases with the circuit size increase. A similar trend is observed for the *SLL* locked circuits, and shown in Figure 3.7(b). As for the *MR* shown in Figure 3.7(c), we observe a negative slope of 0.42. A similar trend is found for *SLL* locked circuits and shown in Figure 3.7(d).

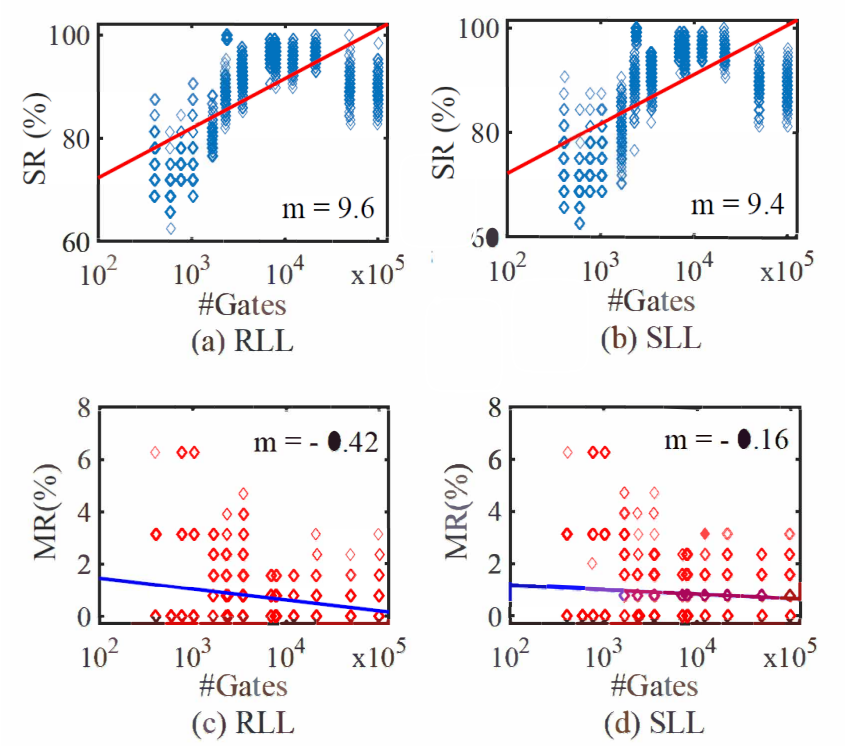

Figure 3.7: Scatter plots of *SR* and *MR* versus number of gates on *RLL* and *SLL* benchmark circuits. (a) *SR* for *RLL* circuits, (b) *SR* for *SLL* circuits, (c) *MR* for *RLL* circuits, and (d) *MR* for *SLL* circuits.

From this observation, we can conclude that the misprediction rate decreases with the increase of the circuit size. Overall, the accuracy of the proposed *TGA* increases for larger circuits.

Table 3.2: *SR* and *MR* for estimating keys for locked circuits from Trust-Hub.

Benchmark	**# Total Gates**	**# Key Inputs**	***Success Rate*(%)**	***Misprediction Rate*(%)**
c880-SL320	404	32	87.50	3.13
c1350-SL320	593	32	78.13	0.00
c1908-SL320	768	32	84.38	3.13
c2670-SL320	1042	32	84.38	3.13
c3540-SL640	1546	64	82.81	1.56
c5315-SL640	2090	64	87.50	1.56
c6288-SL1280	2603	128	96.88	0.00
c7552-SL1280	3173	128	88.28	0.78

To reinforce our conclusion from Table 3.1, we also selected 8 different benchmark circuits from trust-Hub [110] and performed our *TGA* to evaluate the effectiveness. Table 3.2 presents the obtained results for the same. The selected benchmark is noted in Column 1 with the corresponding number of logic gates in the circuit shown in Column 2. Column 3 presents the number of key inputs instead of key gates for each benchmark circuit as one key input may be fed into multiple key

gates. The resultant *SR* and *MR* are concluded in Columns 4 and 5. For c3540-SL640, 64 key inputs are inserted into the circuit. The *SR* is 82.81%, which depicts 53 key inputs that can be predicted with correct values. When comparing the results, 1 incorrect prediction is found, which produces a misprediction rate *MR* of 1.56%. As for the c6288-SL1280 benchmark circuit, 128 key inputs are inserted to protect the circuit. The *SR* of 96.88% can be observed which indicates that the majority of key inputs can be recovered based on our attack (125 key bits). The *MR* is 0.00%, which 0 key bit is mispredicted out of the entire 128 key inputs. We have emphasized *c6288* benchmark circuit to present a clear comparison with the SAT attack, which was not efficient on this circuit.

Besides the evaluation of self-created instances and provided benchmark circuits, we are always interested in attending different global-wise hardware security competitions. We are so proud that we won the first prize for both *UF/FICS Hardware De-obfuscation Competition* (2019) [161] and *HeLLO: CTF '21, Attacks on Hardware Logic Locking & Obfuscation Capture The Flag* (2021) [162] by using our proposed attack.

Note that an adversary can recover the complete key from Table 3.1 with the help of an oracle (e.g., an unlocked chip). As the objective of logic locking is to modify the input-output response of a circuit, it produces incorrect responses for applying the wrong key. If an adversary finds out the key bit location (the unspecified, X, key bits from Algorithm 1, it is easy to determine its value by comparing it with an oracle). As there are few Xs, their permutations are limited and can easily be determined. Note that this could have a complex problem if we do not know the location of the wrong key bit(s). Then, the adversary needs to verify ${}^{|K|}C_N \times 2^N$, where N is the number of unspecified keys, and $|K|$ is the key size. These N unspecified key bits can come from any location of the key K. On the contrary, we only need to verify 2^N cases to determine the complete key, which is a much simpler problem.

3.5.2 Complexity Analysis

SAT problem is an NP-complete problem, thus solving an SAT-resistant locking leads to exponential worst-case complexity. However, our proposed topology-guided attack does not need to compare any input and output pairs, and all the inserted key gates are analyzed individually. Therefore, the time complexity of the attack itself is simply linear to the key size, namely, $O(|K|)$.

Note that, our attack algorithm is based on *FS*, the actual overall complexity is $O(|K| * n * u)$ where n and u represent the size of the netlist and maximum size of the *UFs*, respectively. Thus, the complexity could be considered linear for a particular circuit, since the netlist size is fixed, and the size of *UF* normally ranges from 3-10 gates, depending on the key gate location. In Algorithm 2, once a key bit is predicted and written in the key list K_P, it will never be analyzed again as the value is recovered already. As a result, the computation complexity of launching the attack on *SLL* is the same as it is for *RLL*.

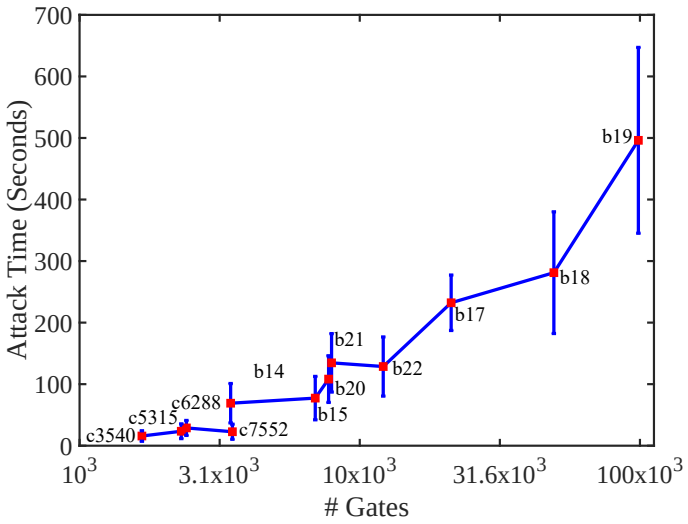

Figure 3.8: Attack time for different *SLL* locked benchmark circuits with 128-bit keys . Each benchmark circuit is evaluated with 100 instances.

The attack time of the proposed *TGA* on different *SLL* locked circuits is illustrated in Figure 9. The plot shows the attack time versus the number of gates for the 100 instances of each benchmark circuit (same circuits mentioned in Table 1). Note that we consider benchmark circuits inserted with 128 key gates only for uniformity. The minimum, average, and maximum values of the attack time are displayed for each benchmark circuit. For example, the reported time for performing *TGA* on 128-bit *SLL* locked b17 benchmark circuit lies in a range from 178 seconds to 277 seconds with an average of 222.38 seconds. The location of inserted key gates causes this variation, and the corresponding number of *EUFs* searched during the attack. The x-axis is presented in the log scale to display all the different benchmarks clearly. The graph displays an exponential distribution, and thus, it can be concluded that the attack time increases linearly with the increase in circuit size.

3.6 Summary

In this chapter, we have discussed a novel oracle-less topology-guided attack based on unit function search. Due to the repetitive deployment of *UF* in a netlist, the key bits for a locked unit function can be determined by constructing *EUFs* with hypothesis key bits and comparing them with the corresponding unlocked *UFs* as self-referencing. Compared to the traditional SAT-based attacks, our proposed *TGA* does not require any input/output pair or an activated chip from the market. Moreover, SAT-resistant countermeasures cannot prevent an adversary from launching this attack. To demonstrate the performance of this attack, we presented the results on different benchmark circuits locked with random logic locking and strong logic locking techniques. We also validated our proposed attack on existing locked benchmark circuits from the trust-Hub. The success rate and misprediction rate metrics are proposed to evaluate the effectiveness of this attack. It is important to emphasize the complexity of this attack which is linear with the key size on both *RLL* and *SLL*, which makes it very effective for circuits with larger key sizes. A countermeasure is also proposed as a solution to prevent this topology-guided attack. The basic idea is to insert the key gate in a unique unit function or lock all the instances repeated in the netlist. Note that this solution can only be used to prevent this topology-guided attack. To design a secure logic locking technique, one needs to select an existing secure logic locking technique along with our proposed solution.

Chapter 4

CamSkyGate: Camouflaged Skyrmion Gates for Protecting ICs

4.1 Introduction

ICs with CMOS technology are approaching the limit of quantum-mechanical boundaries with increased power dissipation and process variation [163]. With the urgent requirement for developing an alternative solution, spintronic devices offer a feasible choice for post-Moore devices, and magnetic skyrmion offers an ideal platform for implementing various designs [164]. Over the past years, different types of skyrmion-based logic gates have been proposed. The effects of skyrmion movement such as skyrmion Hall effect [96, 97], skyrmion-edge repulsion [98], and spin-orbit torque-induced motion [99, 100] were exploited to implement logic functions. With the advantages of minimal power consumption and small device size, the skyrmion-based circuit becomes a competitive candidate beyond traditional CMOS technology [105, 112, 165–167]. However, the security aspects of the skyrmion designs are yet to be explored.

Reverse engineering is a popular technology by which an IC is examined to gain a full understanding of its construction or functionality. Generally, it can be employed by the semiconductor manufacturer to perform failure analysis, defect identification, and verify IP infringement [45, 46, 168]. Unfortunately, the same RE can be exploited by an adversary to reconstruct the gate-level netlist from a chip [88]. As a result, an adversary can clone an entire chip or pirate the extracted netlist. Note that a cloned chip may also be maliciously modified with a hardware Trojan, which can be exploited while the chip is in the field. In the case of a successful RE being performed and a cloned system being constructed with the same functionality as the original one, the attacker has full control of the design and could make huge money profits without the cost and effort of the design and research.

IC camouflaging can be an effective technique to prevent RE so that an adversary cannot extract the inner details of a circuit. In camouflaging, the layout of a gate can be designed in such a way that multiple gates can be mapped to the same layout. Over the years, researchers have proposed different solutions for IC camouflaging. Rajendran et al. [47] proposed camouflaging by creating standard cells with "dummy contact". Erbagci et al. [48] proposed to camouflage a gate based on the utilization of transistors with different threshold voltages. These threshold voltage-defined logic gates are one-time programmed as different functions but with an identical layout. Yasin et al. [95] proposed an IC camouflaging scheme by toggling the output for one minterm of the perturbed function, and a separate camouflaged block is exploited to restore the perturbed minterm. Li et al. demonstrated two camouflaging strategies (low-overhead camouflaging cell generation and AND-tree camouflaging) to realize exponentially increased security levels with a cost of linearly increasing performance overhead [89]. Shakya et al. [49] proposed to add always-on or always-off transistors by doping modification and dummy contacts. As a result, the physical layout of the camouflaged cells is identical to normal logic gates.

Boolean Satisfiability (SAT)-based attack [138] originally proposed to break logic locking [76] can be used to break IC camouflaging very effectively. The attack needs preprocessing of the camouflaged design to convert to a locked design with a secret key. For example, a camouflaged gate, which can be of AND, OR, NAND, or NOR gate, needs to be replaced with four gates and a 4-to-1 multiplexer with two selection inputs. These section inputs are treated as the secret key. An SAT solver will calculate the Distinguishing Input Patterns (DIPs) and help eliminate the wrong keys. When the correct key is recovered, all the multiplexers will be replaced with the corresponding logic gate. Thus, a secure camouflaging scheme must be SAT-resilient. The covert gate design proposed in [49] can prevent SAT attacks for CMOS designs. However, no solution has been proposed so far for spintronic devices, such as magnetic skyrmions-based circuits.

4.2 Background

4.2.1 Skyrmion Nucleation and Detection

Skyrmion is a magnetic texture protected by topology and behaves as a stable pseudoparticle. In logic device applications, a nanotrack with heavy metal (HM) and ferromagnetic (FM)/perpendicular magnetic anisotropy (PMA) bilayer is usually used to house a skyrmion. Meanwhile, a magnetic tunnel junction (MTJ) is fabricated on top of the nanotrack for the nucleation of skyrmions [101]. In the nucleation process, a local spin-polarized current follows through MTJ and flips part of the magnetization in the PMA layer, which can form a skyrmion if adequate Dzyaloshinskii–Moriya Interaction (DMI) is present in the nanotrack [169, 170]. The skyrmion detection can also be realized through a readout MTJ. The tunneling magnetoresistance (TMR) can be dictated by the skyrmion appearance. The presence (absence) of skyrmion underneath the MTJ corresponds to a high (low) TMR value, which can represent logic 1 and logic 0, respectively [171]. The output signal can cascade directly to the gate inputs, and be synchronized through VCMA [172].

4.2.2 Skyrmion Movement

Skyrmion moves through a structure called nanotrack, made of an FM layer and an HM layer [102]. The HM layer has a sidewall-like structure that wraps the FM layer at the bottom and on two sides. The skyrmion stays at the FM/HM interface. The sidewall wrapping structure eliminates the transverse motion of the skyrmion caused by the skyrmion Hall effect, allowing only linear motion. In order to drive the skyrmion in the nanotrack, a continuous electric current J is required in the HM layer. Due to the spin Hall effect, J generates a spin current J_{s} in the FM layer. At the FM/HM interface, the spin current applies a spin-orbit torque on the skyrmion, driving it along the y-axis, while the Hall effect tends to move the skyrmion transversely along the x-axis. The dynamics of a skyrmion is governed by the Landau–Lifshitz– Gilbert–Slonczewski (LLG) equation:

$$\frac{dm}{dt} = -|\gamma| m \times H_{eff} + \alpha(m \times \frac{dm}{dt}) + \tau_{SOT} \tag{4.1}$$

where m is the normalized magnetization M/M_{s}. M stands for magnetization, M_{s} is the saturation magnetization and H_{eff} is the effective magnetic field associated with magnetocrystalline anisotropic energy and the DMI energy. Further, γ is the gyromagnetic ratio, α is the damping coefficient, and τ_{SOT} represents the spin-orbit torque determined by multiple parts: a gyromagnetic ratio, effective field spin polarization rate, permeability of vacuum, driving current density, and the thickness of magnetic film.

The skyrmion inside a nanotrack is driven by a current flowing in the HM layer via spin orbit torque (SOT). The forces on the micromagnetic skyrmion can be modeled by Thiele equation [173]:

$$G \times v - \alpha D + F_{SOT} - \nabla V = 0 \tag{4.2}$$

The first term describes the Magnus force, in which G presents the gyromagnetic coupling vector, and v is the skyrmion velocity. Dissipative force is represented by the multiplication of damping coefficient α and the dissipative tensor D. The third term represents the driving force F_{SOT} generated by the spin Hall effect. The last term shows the resultant force on the skyrmion, and V presents the confining potential due to boundaries, process impurities, and other textures.

4.2.3 VCMA Effect

Skyrmion synchronization is a critical requirement when considering the functionality of a scalable skyrmion system. As the skyrmion gates work based on the skyrmion-skyrmion interaction, it is necessary that different skyrmions arrive at the inputs at the same time. In other words, the skyrmions need to be held at the inputs of the gates. The authors in [172] proposed a Voltage-Controlled Magnetic Anisotropy (VCMA) based synchronizer at the input of the gate so that skyrmions cannot enter inside the nanotrack. A clock notch [105, 165] can also be placed instead of a VCMA to synchronize the skyrmion movement. We adopt this VCMA to control the skyrmion movement as it provides better controllability.

To vary the uniaxial anisotropy of a magnetic material, the VCMA technique can be exploited by providing an applied electric field. Once a voltage is applied to cross the ferromagnetic nanotrack, the electron density will be changed which will change the perpendicular magnetic anisotropy

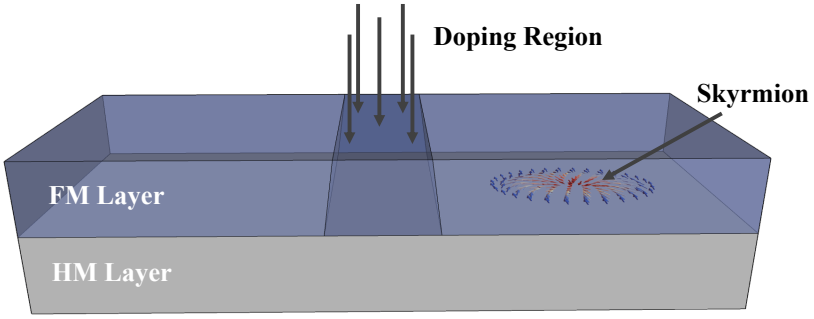

Figure 4.1: An abstract view of doping process at the FM layer.

in turn. It is noticed that the changed anisotropy is predominantly linear to the applied voltage, and the equation can be concluded as follows:

$$K_{\mu v} = K_{\mu} + \zeta E_b \tag{4.3}$$

where $K_{\mu v}$ is the resultant anisotropy in the affected region, and K_{μ} presents the value of original anistropy. The electric field is present by E_b, and ζ is the coefficient of the VCMA effect. When enough positive voltage is given, the skyrmions will be stopped at the affected region. When no electrical field is applied, the skyrmions will move in the nanotrack normally.

4.2.4 Doping Effect

Selective doping can be used to modulate magnetic anisotropy in a local area in magnetic thin films. When the magnetic anisotropy is significantly different from the rest of the thin film, the skyrmion propagation can be blocked. A common method to realize selective doping is ion implantation [170, 174–176]. It emits an energetic ion beam to dope foreign ions (e.g., Ga^+ or Ar^+) to a magnetic thin film. Thus, it is vastly used to modulate the anisotropy of magnetic thin films. In the past, it has facilitated the fabrication of bit patterned media for magnetic recording, and fabrication of nanomagnetic logic systems [174, 177]. In the devices we have proposed in this article, ion implantation is a viable technique to realize doping in a local region to control skyrmion propagation, which is shown in Figure 4.1. Note that doping is a non-reversible process and will modify the magnetic anisotropy permanently. However, the magnetic anisotropy in the VCMA region can be tuned by applying a different voltage.

4.2.5 Simulation Parameters

In the dissertation, the micro-magnetic simulations were performed using $\mathbf{mumax^3}$ tool, which is a GPU-based accelerated program that can analyze the dynamic behavior of skyrmions. The movement of a skyrmion in the track is modeled based on Equation 4.2 where the electrical current in the HM layer drives the skyrmion. Parameters used in simulation are: Gilbert damping factor $\alpha = 0.3$, non adiabatic STT factor $\beta = 0.1$, exchange stiffness $A_{\text{ex}} = 1.5 \times 10^{-13}$ J/m, perpendicular magnetic anisotropy $K_{\text{u}} = 6 \times 10^5$ J/m^3, saturation magnetization $M_{\text{s}} = 5.8 \times 10^5$ A/m, and DMI constant $D = 3 \times 10^3$ J/m^2. Mesh sizes are 1 nm $\times$ 1 nm $\times$ 0.4 nm, along the X, Y, and Z axes. In our proposed CamSkyGate designs, we create differential doped regions, where the heavily doped region blocks the propagation of a skyrmion, and the lightly doped region does not affect the skyrmion movement (see the details in Section 4.3). We simulate the heavily doped region with parameter $1.2 \times K_{\text{u}}$ and the lightly doped region with $1.1 \times K_{\text{u}}$. The study from Se Young Park et al. [178] showed the modulation of magnetic anisotropy by the changing of chemical potential. Therefore, the K_{u} can be modulated using the doping method. The choice of these parameters allows the realization of the camouflaged gate.

4.3 Proposed CamSkyGate Design

This section introduces our proposed designs for gate camouflaging [112]. Since IC camouflaging aims to protect threats from reverse engineering, we will start describing this section with the adversarial model first.

4.3.1 Adversarial Model

Generally, a secure logic camouflaging relies upon the fact that an adversary cannot determine the actual gate-level netlist from a camouflaged design. In the attack model, we treat the foundry as trusted, and no attack will be performed at the manufacturing site, which is similar to prior camouflaging schemes. The adversary can be any entity in the supply chain or market other than the foundry that has the capability of performing RE.

- Camouflaged gate-level netlist reconstruction: The adversary can acquire a working chip from the market and has the capability to obtain SEM images by delayering the chip. The camouflaged netlist can be constructed from these SEM images.
- Internal scan access: The adversary has access to the scan design so that it can perform SAT attack [138]. One can also assume that the adversary can not access the internal scan chains and launch a sequential SAT attack. However, if we show the camouflaged design is secure against SAT, then automatically, it will be secure against sequential SAT attacks. As a result, this study assume that the adversary has scan access.

4.3.2 CamSkyGate Design Principles

The camouflaged skyrmion gates operate based on the skyrmion-skyrmion interaction, similar to the traditional skyrmion gates. The additional modification that leads to the camouflaging is from different doping regions that look the same as the other regions. This section introduces the primary design principles to build a CamSkyGate.

1. *Skyrmion Motion Control:* The skyrmion motion can be controlled by applying doping at different regions of the nanotrack. The perpendicular magnetic anisotropy of the doping region can be changed so that it affects the motion of the skyrmions. The doping region blocks the skyrmion to move into the annihilation region of the nanotrack.
2. *Topological Indistinguishability*: The CamSkyGate uses the selective differential doping technique to implement different logic functions from the same layout. While doping has a strong influence on the skyrmion propagation, SEM cannot distinguish regions with different doping concentrations if they are designed properly, as demonstrated by Frank et al. [179]. The results showed that regions with different doping levels with selected doping concentrations have little effect on the contrast of the SEM images. Thus, the different doping regions of CamSkyGate cannot be identified by performing RE and only identical layouts will be recovered.
3. *Annihilation of Redundant Skyrmions*: The layout of camouflaged gates is designed in such a way that more than two gate functionalities can be obtained simultaneously. For example, the AND and OR gates can use the same layout depending on the doping regions (see Figure 4.2).

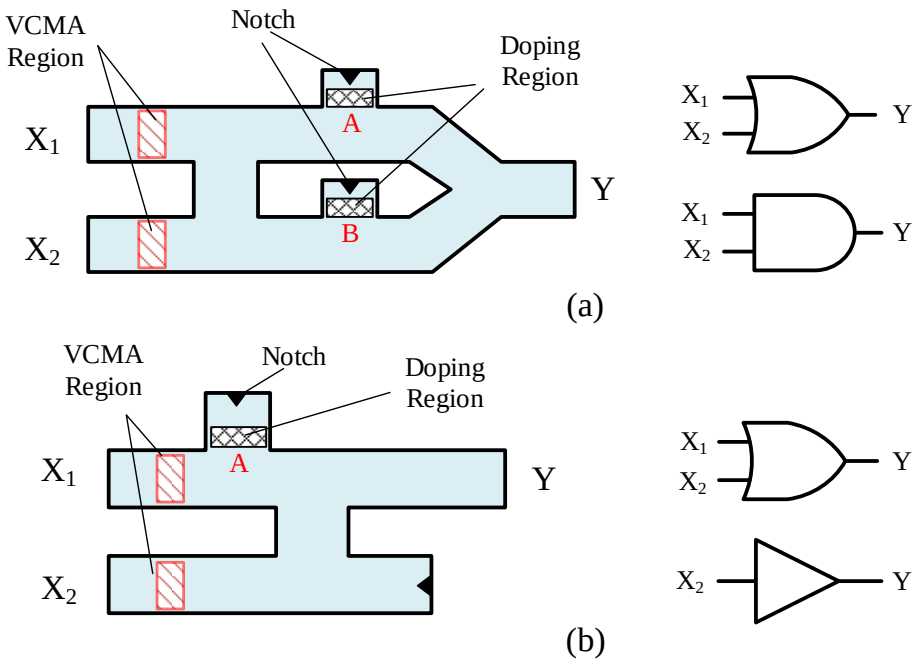

Figure 4.2: Simple camouflaged skyrmion gates. (a) AND and OR CamSkyGate. (b) OR and BUF CamSkyGate.

It is thus necessary to annihilate one or more skyrmions from the nanotrack; otherwise, multiple skyrmions will arrive at the gate output.

In the following, we will present different simple and complex designs of CamSkyGates.

4.3.3 Simple Camouflaged Gates

Figure 4.2 shows two simple camouflaged cell designs with two inputs (X_1 and X_2) and one output (Y). Figure 4.2.a shows the layout of a camouflaged cell to implement AND and OR functions. Two VCMA regions at the input of the gate are placed to synchronize the motion of the skyrmions at each nanotrack. Two doping regions, denoted A and B, highlighted in black and white stripes, are selected for doping to determine the function of the cell. When region A is heavily doped with gallium (Ga^+) ions, its magnetic anisotropy changes, thus, region A can block a skyrmion from entering the notch region in the top nanotrack. Region B will be lightly doped. The layout will result in the OR function. When region B is heavily doped with gallium ions, it will prevent the skyrmion on the bottom track to enter the annihilation region, and region A will be lightly doped. At this point, the cell will behave as an AND gate. We propose to use differential doping to make the two regions the same under SEM images [179]. One region is

heavily doped to prevent the skyrmion motion and the other region is lightly doped so that it does not impact the skyrmion propagation. The heavily and lightly doped regions result in $1.2 \times K_{\mathrm{u}}$ and $1.1 \times K_{\mathrm{u}}$, respectively. The layouts of these gates are perfectly symmetrical, and it is infeasible for an adversary to determine the functionality using image analysis.

Figure 4.2.b illustrates the overall structure of the camouflaged cell for an OR gate and a BUF. Unlike the AND/OR CamSkyGate, one doping region is sufficient for determining the OR and BUF functions. When region A is doped to achieve $1.2 \times K_{\mathrm{u}}$, the OR function will be realized as the skyrmion on the upper track cannot enter the annihilation region. In the case the region is lightly doped to obtain $1.1 \times K_{\mathrm{u}}$, the skyrmion on the top nanotrack will pass the doping region and be annihilated at the notch. This makes input X_1 redundant, and it becomes a dummy connection. Due to the absence of a skyrmion at the upper nanotrack, a skyrmion at input X_2 will reach the output Y. As a result, a BUF will be realized.

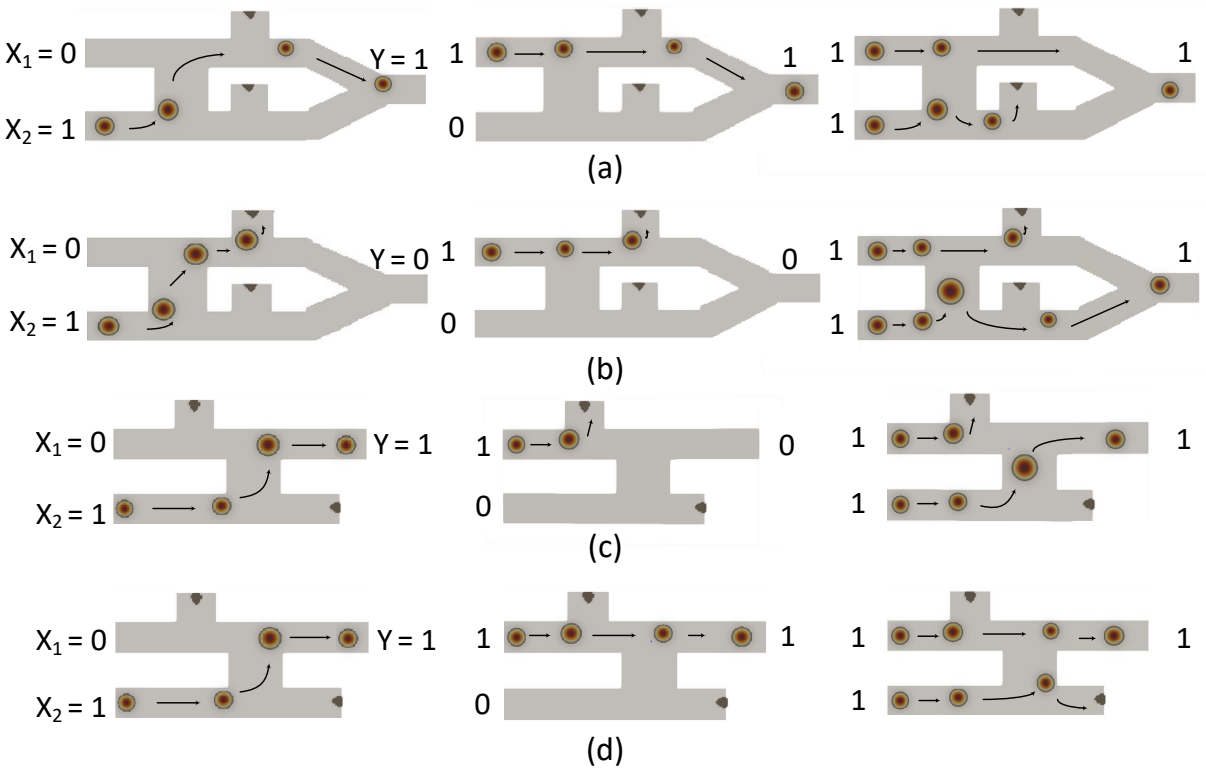

Figure 4.3: Simulation results for different simple CamSkyGates. (a) OR gate, (b) AND gate, (c) BUF, and (d) OR gate.

Figure 4.3 shows the simulation results of different simple CamSkyGates. We use the GPU-based accelerated $\mathbf{mumax^3}$ tool to simulate all the gates. Note that logic 1 and logic 0 are realized using

the presence and absence of a skyrmion, respectively. The simulations for all the inputs with logic 0 are redundant as there are no skyrmions in any of the nanotracks of a gate. In the figure, the motion trajectory of a specific skyrmion is illustrated by arrows. Figure 4.3.a and Figure 4.3.b show the simulation results for OR/AND CamSkyGate with different input combinations. For input $X_1X_2 = 01$, the skyrmion from the lower track moves to the upper track and reaches the output (Figure 4.3.b) while it gets destroyed at the annihilation region (Figure 4.3.b). Note that region A (see Figure 4.2.a) in the upper nanotrack is heavily doped while region B is lightly doped for an OR gate and vice versa for an AND gate, the skyrmion at the upper nanotrack can pass successfully reach to the output for an OR gate while it is destroyed at the annihilation region for an AND gate. Similar analysis can be done for other input combinations. Figures 4.3.c and 4.3.d shows the simulation results BUF and OR functions. When region A is lightly doped, the skyrmion on the upper nanotrack moves to the annihilation region and gets destroyed. This results in the input X_1 being redundant, and a BUF is realized. Region A is heavily doped for an OR gate, and no skyrmion can enter into the annihilation region. One can find all the simulation videos for each CamSkyGate with all input combinations in [180].

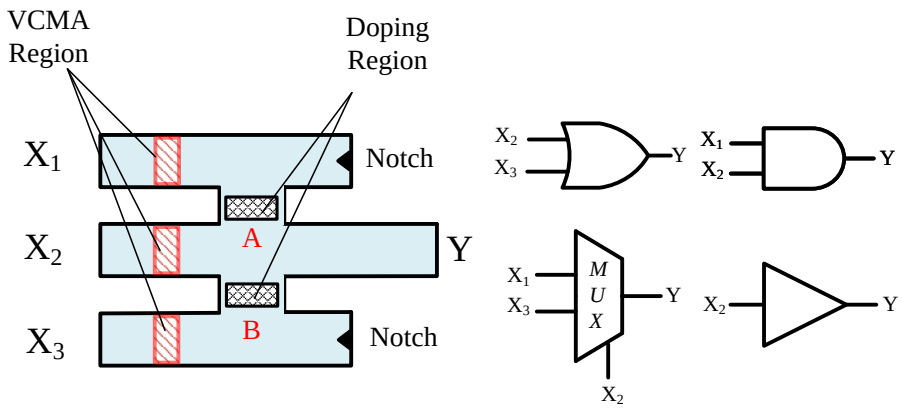

Figure 4.4: Complex CamSkyGate design with different functions between OR, AND, MUX, and BUF.

4.3.4 Complex Camouflaged Gate

Besides the simple CamSkyGate designs with two inputs, we have also proposed a complex CamSkyGate design with multiple inputs which is shown in Figure 4.4. The layout for a two-input

OR gate, two-input AND, 2-to-1 MUX, and a BUF is shown in Figure 4.4. The doping regions, A and B, are highlighted as black and white stripes like before. Region A is inserted between the upper two nanotracks while B is located between the lower two nanotracks. When A is heavily doped and B is lightly doped, it will block the movement of skyrmions from the lower nanotracks to the upper one. Therefore, X_2 and X_3 will determine the gate functionality, and X_1 becomes the dummy input. If A is lightly doped and B is processed with heavy doping, the lower nanotrack will be blocked and a AND gate can be implemented. X_1 and X_2 are the inputs and determine the functionality while X_3 is the dummy input. When both regions A and B are lightly doped, the skyrmions from all the nanotracks will interact, and a 2-to-1 MUX function will be obtained, where X_2 becomes the selector input and X_1 and X_3 are the inputs. When there is a skyrmion present at X_2 input (i.e., X_2 is at logic 1), the output Y will be decided by input X_1, otherwise by input X_3. In the case of A and B are all heavily doped, there is no interaction between each nanotrack and the CamSkyGate can realize the function of a simple buffer in which X_2 is the input. Both X_1 and X_3 will be the dummy inputs in this layout of CamSkyGate.

Figure 4.5 shows the simulation results for the complex CamSkyGate presented in Figure 4.4 which implements four different functions. Figure 4.5.a shows the simulation for a two-input OR gate. As input X_1 is a dummy connection, it will have no impact on the gate functionality. The input pattern $X_1X_2X_3 = [100]$ results $Y = 0$ which validates the effect of X_1 on Y. All other combinations $X_2X_3 = [10, 01, 11]$ results $Y = 1$. Figure 4.5.b shows the simulation for a two-input AND gate with X_3 as a dummy input. We apply $X_3 = 1$ to verify that it has no impact on Y. All three input pattern $X_1X_2 = [00, 01, 10]$ results $Y = 0$ and one input $X_1X_2 = [11]$ makes $Y = 1$ which validates the AND function. The simulations for 2-to-1 MUX are illustrated in Figure 4.5.c. When the selection input $X_2 = 0$, it selects $Y = X_3$ otherwise, $Y = X_1$. As a result, $Y = 0$ when $X_1X_2X_3 = [100, 011]$ and $Y = 1$ when $X_1X_2X_3 = [001, 110]$. Finally, Figure 4.5.d shows the simulation for a buffer with X_1 and X_3 as dummy inputs. The output Y becomes logic 1, when input X_2 is logic 1.

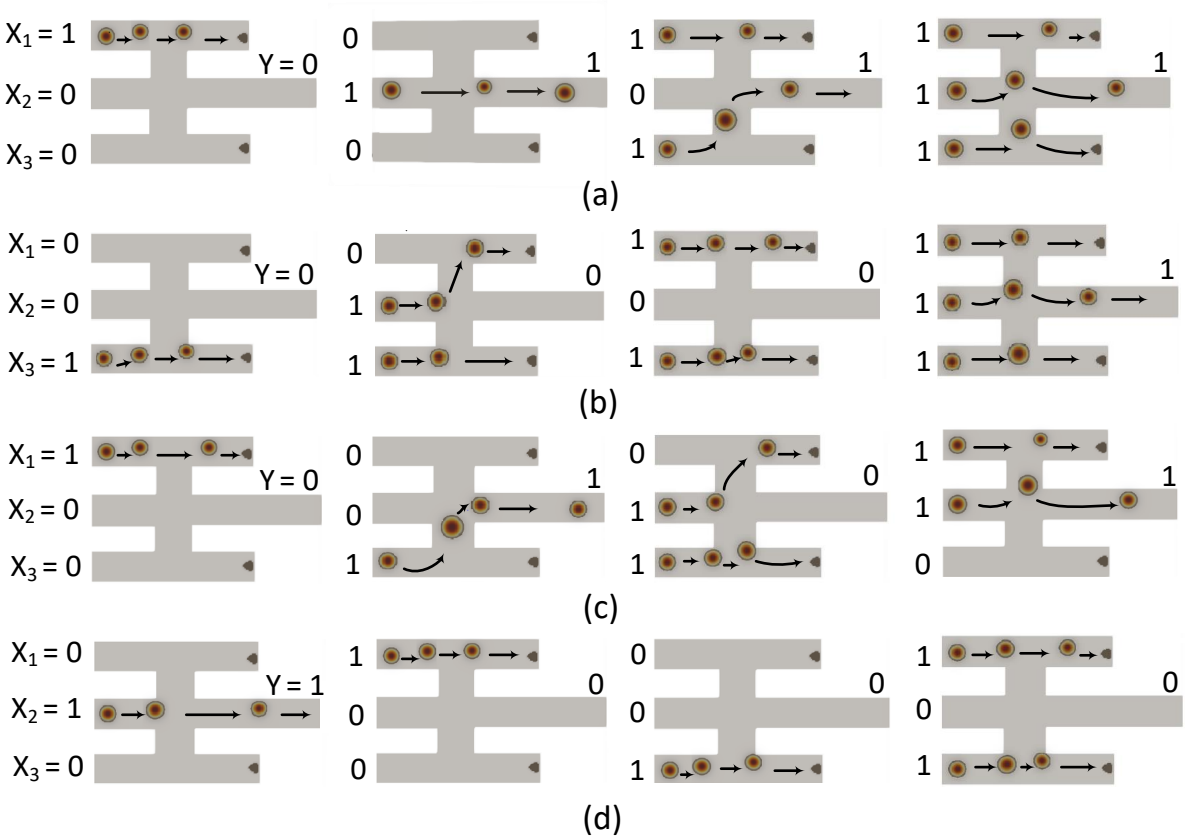

Figure 4.5: Simulation result examples for different complex gate designs. (a) Two-input OR gate, (b) Two-input AND gate, (c) 2-to-1 MUX, and (d) BUF.

4.4 Security Analysis

This section evaluates the security of our proposed design against the SAT attack [138]. We also provide a detailed comparison with the existing CMOS-based camouflaging to show the effectiveness of our proposed CamSkyGate design for emerging circuit camouflaging.

4.4.1 SAT-based Attack on Camouflaged Circuits

The SAT attack is an effective way to determine the logic function of a camouflaged gate. An adversary needs to perform reverse engineering to obtain the gate-level netlist. As the logic function of each CamSkyGate is unknown, it needs to convert the camouflaged gate with its key-based equivalent [181]. For example, the simple AND/OR CamSkyGate, shown in Figure 4.2.a, can be replaced with a MUX where the key bit at the selector input selects one of the two gate combinations. Similarly, the complex CamSkyGate, shown in Figure 4.4, can be replaced with

a MUX, where two key bits at the selector input select one of the four gate combinations. An adversary can reconstruct a key-based netlist from the camouflaged circuit and apply the SAT attack to determine the key. Once the value of the key is determined, he/she can identify all the camouflaged gates and construct the original gate-level netlist.

Since the function of our designed camouflaged skyrmion cell cannot be determined by RE or imaging processing, the attacker needs to replace all the CamSkyGates with the MUX-based selection networks. Table 4.1 shows the SAT attack resistance evaluation for different circuits using our proposed CamSkyGate camouflaging scheme. Five benchmark circuits from ISCAS'85 [108] and two circuits from EPFL suite [182] are selected to test the effectiveness and listed in Column 1. Column 2 represents the gate count for each benchmark circuit. To perform the SAT attack, we camouflage each benchmark circuit by placing complex CamSkyGate and simple CamSkyGate with a ratio of 1:2 and replacing them with the corresponding MUX structure for performing the SAT attack. Since the complex CamSkyGate requires two selection bits for the selection network while the simple design requires one, the exact key space size is identical for both complex and simple CamSkyGates. We perform the SAT attack using the code provided in [30] and compare the performance with the results provided in [49] (Regular Camouflaging and Covert Gate Camouflaging in Table 4.1). Timeout for the attack was also set to 12 hours, which is in line with the prior work [30, 49, 183]. For each camouflaging scheme, we have shown the size of the key (Columns 3, 6, and 9), the attack time (Columns 4, 7, and 10), and the number of iterations to launch the attack (Columns 5, 8, and 11). Since the longer key value will lead to higher overall search space for the SAT solver, the run time of the SAT attack will also be increased. In the security evaluation, we determine the unit of run time is in hours which is the same as the unit of covert gate camouflaging evaluation presented in [49], while the evaluation on regular camouflaging provided in [49] is on the scale of seconds. The camouflaging scheme presented here and in [49] can not provide security for the c1908 benchmark circuit due to its small size. The SAT attack becomes ineffective for all the remaining benchmarks (i.e., a timeout that is over 12 hours). As a result, it can be concluded that our proposed CamSkyGate design could provide the same level of security compared with the CMOS-based camouflaging scheme [49], which indicates the feasibility of protecting the IP privacy of the skyrmion-based circuit.

4.5 Summary

In this chapter, we have proposed several novel skyrmion-based camouflaged gates denoted as CamSkyGate to protect a spintronic design from reverse engineering. We apply differential doping at the different regions on a camouflaged layout to implement different logic functions. Doping can change the physical parameters (e.g., magnetic anisotropy K_{u}) and control the skyrmion motion. A region with light doping enables the skyrmion pass through the nanotrack, whereas the heavily doped region will block its propagation. Different gate functionality can be obtained in a single layout by placing these heavily and lightly doped regions to control skyrmion-skyrmion interaction. As it is infeasible to distinguish these heavily and lightly doped regions, the functionality of a CamSkyGate cannot be determined using SEM imaging which is commonly used for reverse engineering. The functionality of each CamSkyGate is simulated using the $\mathbf{mumax^3}$ simulation tool. To further launch the attack and recover the full functionality of the entire circuit, the adversary is required to synthesize the skyrmion-based circuit into a gate-level netlist and construct a MUX-based network for each camouflaged cell. The SAT-based security evaluation shows that our proposed design can provide the same level of protection similar to the traditional CMOS-based camouflaging.

Table 4.1: Comparison of SAT attack resiliency.

Benchmark	Gate Count	Regular Camouflaging [49]			Covert Gate Camouflaging [49]			Proposed Camouflaging		
		5% of NAND/NOR/XOR			NAND + NOR + AND + OR			AND+OR+MUX+BUF		
		|K|	Attack Time (s)	# iterations	|K|	Attack Time (hrs)	# iterations	|K|	Attack Time (hrs)	# iterations
c1908	880	34	0.55	7	811	3.25	235	756	3.44	217
c2670	1193	26	0.65	11	1514	Timeout	2127	1040	Timeout	430
c3540	1669	28	0.68	11	2088	Timeout	28	1488	Timeout	180
c5315	2307	46	3.58	25	3379	Timeout	24	1774	Timeout	305
c7552	3512	106	4.07	27	4454	Timeout	52	2000	Timeout	87
arbiter	11839	1182	3815	855	23678	Timeout	82	4000	Timeout	3490
voter	13758	1078	Timeout	33	21560	Timeout	51	4000	Timeout	67

Chapter 5

Conclusion and Future Work

This chapter provides the summary of this dissertation and some suggestions for future work.

5.1 Conclusion of Dissertation

The rise of recycled ICs in critical infrastructures causes a major concern to the government and industry because these chips exhibit lower performance and shorter remaining lifetimes. Furthermore, the outsourcing of the design and manufacturing of ICs in the current horizontal semiconductor integration flow has posed various security threats due to the presence of untrusted entities, such as overproduction of ICs, the sale of out-of-specification/rejected ICs, and piracy of IPs. Over the years, researchers have continually proposed different solutions to enable trust in digital ICs. However, there are still different issues left that need to be addressed. First, it is generally necessary to power up the chip and collect the parameters to determine the prior usage [63, 67, 71, 184–189]. However, the distributors in the supply chain may not test the chips properly without expensive test infrastructure. Also, the test with powering up may cause extra defects on authentic chips. Second, logic locking has become a well-accepted solution due to its simple structure and low overhead. A circuit will be locked with a secret key which is generally stored in a tamper-proof memory. The locked circuit will work functionally only if the correct key is applied. SAT-based attacks have been shown to efficiently compromise key-based obfuscation methods with a rather small number of distinguishing input patterns (DIPs). However, there is not much research on attack and countermeasure focusing on the structure of the design. Third, reverse engineering is also considered a powerful attack to threaten the IP. The attacker can launch the

attack and recover the full function of the design. IC camouflaging has been proposed to protect the traditional CMOS device against RE. However, the investigation of camouflaging on emerging technology (e.g., skyrmion-based circuits) has not been explored. In this dissertation, we have considered and addressed the mentioned issues to enable trust in digital ICs.

In Chapter 2, we have addressed the trust problems in the supply chain management from manufacturer to system integrator through multiple distributors. A solution for enabling end-to-end traceability against recycled ICs in the semiconductor supply chain is proposed. Instead of storing the information on the chip. An RFID tag is utilized to empower the distributors to verify the information without powering up the chip. Any tampering with RFID tag memory content can be detected and located. The initial frequency data provided by the manufacturer needs to be verified by the system integrator for the final decision of whether a chip has been recycled or not. The system integrator will power up the chip and verify the prior usage of the IC.

In Chapter 3, we described a novel oracle-less attack on logic locking based on self-referencing. It relies on identifying repeated functions for determining the value of a key bit. Our proposed graph search algorithm can efficiently find a duplicate function of the locked part of the circuit, and it only takes a few seconds to determine the key bit. The attack performance is evaluated on both our created instances and locked benchmark circuits from TrustHub. The majority of keys can be predicted successfully and efficiently. We also present a solution to prevent this topology-guided attack and make logic locking secure.

In Chapter 4, we investigate and explore to design of a secure IC camouflaging technology using emerging technology. We have proposed skyrmion-based gate designs for IC camouflaging against reverse engineering. The same layout can implement various types of logic functions based on applying differential doping techniques. As a result, the attacker cannot extract the full function of a camouflaged skyrmion-based circuit, making it secure against RE. The performance against SAT-attack has also been evaluated, and the design is able to achieve the security level with a smaller key size compared with the traditional CMOS counterpart.

5.2 Future Work

This section discusses possible extensions of the work presented in this dissertation.

5.2.1 Protecting ICs in Supply Chain

The RFID-based system helps us to enable the traceability of ICs in the supply chain and empowers the distributors to authenticate the stored data in the RFID tag without powering up the chip itself. However, the memory size of RFID tags can be a limitation since only one manufacturer and several distributors' data can be stored due to the length of public keys and signatures. In the future, we plan to migrate the data storage from the RFID tag to the blockchain, where end-to-end traceability is to be implemented. Instead, only a single ID of the tag itself with a few bits is necessary to be stored in the tag memory. Different implementation of enabling IC traceability on blockchain are proposed in [190–198]. The combination of an RFID-based system and data management on the blockchain can solve the data management issues and keep empowering the distributors with the same access to authenticate the chip. When the chips are delivered to the distributor/system integrator, it can extract the ID of the tag by using a commercial RFID reader. The related information can be found on the blockchain. A local verification can be performed similarly to the existed infrastructure. Once the signature(s) is(are) valid, the current entity needs to create its own information and update them in the blockchain instead of writing the RFID tag.

5.2.2 Improvement of Topology-Guided-Attack

Our proposed oracle-less attack can predict the majority of the key bits. In the future, we consider applying machine learning to train the model to improve our proposed *TGA*. By applying different machine learning methods, the performance comparison can be performed to select the proper one with better accuracy. Furthermore, machine learning could help calculate the weight of each critical feature during the training process, which could help us improve the attack from different perspectives. Different input vectors are necessary to be constructed and fed into the model as training data (e.g., locked gate-level netlist, locked *UFs* with different layers, and constructed *EUFs*). The correct key lists are also necessary to train the model, which can be obtained by performing our attack or other state-of-art attacks. By providing adequate training data, we expect a better training model which can recover the key list of a locked circuit with

a higher success rate with a lower misprediction rate. We plan to investigate the possibility of compromising our proposed countermeasure by launching machine learning-based *TGA*.

5.2.3 IC Camouflaging on Emerging Technology

Currently, our proposed CamSkyGate with experimental simulation showed the feasibility of implementation, and similar security compared with traditional CMOS technology. In the future, designing complex CamSkyGates with additional types can help us to achieve better security. More dummy connections can also be helpful to increase an attacker's difficulty. The advantages of proposing additional complex CamSkyGate designs can be summarized as follows:

- Better flexibility: Currently, our designed CamSkyGates can replace and camouflage the following gate types: AND, OR, BUF, and MUX. In the future, other gate types (e.g., NAND, NOR, XOR, and XNOR) can also be camouflaged by the new CamSkyGate design, which makes the synthesis process compatible with the traditional CMOS gates.
- Enhanced security: To compromise the key value and extract the full function of the circuit, the attacker needs to convert each CamSkyGate into a MUX-based network and then apply the SAT solver to calculate the key value. Compared with the simple CamSkyGate, the complex CamSkyGate has more gate combinations and includes more function types. Thus, a larger network will be constructed, and more select bits (key-value) need to be considered. Currently, our complex gate needs be to replaced with a 4-to-1 MUX with two key bits and requires more iterations of the SAT solver when compared with a 2-to-1 MUX generated from a simple CamSkyGate. The future work could be designing a complex CamSkyGate which requires a larger MUX network construction when performing the attack. This will raise the difficulty of launching an SAT attack on the camouflaged gates [49].

Bibliography

[1] M. Alam, S. Chowdhury, M. Tehranipoor, and U. Guin, "Robust, Low-Cost, and Accurate Detection of Recycled ICs using Digital Signatures," in *IEEE Int. Symposium on Hardware Oriented Security and Trust (HOST)*, 2018.

[2] U. Guin, D. DiMase, and M. Tehranipoor, "Counterfeit Integrated Circuits: Detection, Avoidance, and the Challenges Ahead," *Journal of Electronic Testing*, vol. 30, no. 1, pp. 9–23, 2014.

[3] U. Guin, K. Huang, D. DiMase, J. M. Carulli, M. Tehranipoor, and Y. Makris, "Counterfeit Integrated Circuits: A Rising Threat in the Global Semiconductor Supply Chain," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1207–1228, 2014.

[4] M. Tehranipoor, U. Guin, and D. Forte, *Counterfeit Integrated Circuits: Detection and Avoidance*. Springer International Publishing, 2015.

[5] D. DiMase, Z. A. Collier, J. Muldavin, J. A. Chandy, D. Davidson, D. Doran, U. Guin, J. Hallman, J. Heebink, E. Hall, Honorable A. R. Shaffer, "Zero Trust for Hardware Supply Chains: Challenges in Application of Zero Trust Principles to Hardware," *National Defense Industrial Association (NDIA)*, 2021.

[6] IHS iSuppli, "Top 5 Most Counterfeited Parts Represent a $169 Billion Potential Challenge for Global Semiconductor Market," 2011.

[7] F. Koushanfar and G. Qu, "Hardware metering," in *Proc. IEEE-ACM Design Automation Conference*, pp. 490–493, 2001.

[8] J. Roy, F. Koushanfar, and I. Markov, "EPIC: Ending Piracy of Integrated Circuits," in *Proc. on Design, Automation and Test in Europe*, pp. 1069 –1074, March 2008.

[9] Y. Alkabani, F. Koushanfar, and M. Potkonjak, "Remote activation of ICs for piracy prevention and digital right management," in *Proc. of IEEE/ACM international conference on Computer-aided design*, pp. 674–677, 2007.

[10] Y. M. Alkabani and F. Koushanfar, "Active hardware metering for intellectual property protection and security," in *Proc. of USENIX Security Symposium*, pp. 20:1–20:16, 2007.

[11] M. Tehranipoor and C. Wang, *Introduction to Hardware Security and Trust*. Springer Science & Business Media, 2011.

[12] E. Castillo, U. Meyer-Baese, A. García, L. Parrilla, and A. Lloris, "IPP@HDL: Efficient Intellectual Property Protection Scheme for IP Cores," *IEEE Trans. on VLSI (TVLSI)*, pp. 578–591, 2007.

[13] S. Bhunia and M. Tehranipoor, *Hardware Security: A Hands-on Learning Approach*. Morgan Kaufmann, 2018.

[14] S. Adee, "The hunt for the kill switch," *iEEE SpEctrum*, vol. 45, no. 5, pp. 34–39, 2008.

[15] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware trojans," *Computer*, vol. 43, no. 10, pp. 39–46, 2010.

[16] M. Tehranipoor and F. Koushanfar, "A survey of hardware trojan taxonomy and detection," *IEEE design & test of computers*, vol. 27, no. 1, pp. 10–25, 2010.

[17] K. Xiao, D. Forte, Y. Jin, R. Karri, S. Bhunia, and M. Tehranipoor, "Hardware Trojans: Lessons learned after one decade of research," *Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 22, no. 1, p. 6, 2016.

[18] A. Jain, Z. Zhou, and U. Guin, "Survey of Recent Developments for Hardware Trojan Detection," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2021.

[19] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *Proc. of ACM/IEEE on Design Automation Conference*, pp. 83–89, 2012.

[20] U. Guin, Q. Shi, D. Forte, and M. Tehranipoor, "FORTIS: A Comprehensive Solution for Establishing Forward Trust for Protecting IPs and ICs," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 2016.

[21] U. Guin, Z. Zhou, and A. Singh, "Robust Design-for-Security Architecture for Enabling Trust in IC Manufacturing and Test," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2018.

[22] U. Guin, Z. Zhou, and A. Singh, "A novel design-for-security (DFS) architecture to prevent unauthorized IC overproduction," in *Proc. of the IEEE VLSI Test Symposium (VTS)*, pp. 1–6, April 2017.

[23] J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri, "Fault analysis-based logic encryption," *IEEE Transactions on Computers*, pp. 410–424, 2015.

[24] S. M. Plaza and I. L. Markov, "Solving the third-shift problem in IC piracy with test-aware logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 961–971, 2015.

[25] Y.-W. Lee and N. A. Touba, "Improving logic obfuscation via logic cone analysis," in *Latin-American Test Symposium (LATS)*, pp. 1–6, 2015.

[26] A. Baumgarten, A. Tyagi, and J. Zambreno, "Preventing IC piracy using reconfigurable logic barriers," *IEEE Design & Test of Computers*, pp. 66–75, 2010.

[27] S. Khaleghi, K. Da Zhao, and W. Rao, "IC piracy prevention via design withholding and entanglement," in *The 20th Asia and South Pacific Design Automation Conference*, pp. 821–826, 2015.

[28] B. Liu and B. Wang, "Embedded reconfigurable logic for ASIC design obfuscation against supply chain attacks," in *Proceedings of the conference on Design, Automation & Test in Europe*, p. 243, 2014.

[29] R. S. Chakraborty and S. Bhunia, "HARPOON: an obfuscation-based SoC design methodology for hardware protection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1493–1502, 2009.

[30] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 137–143, 2015.

[31] X. Xu, B. Shakya, M. M. Tehranipoor, and D. Forte, "Novel Bypass Attack and BDD-based Tradeoff Analysis Against all Known Logic Locking Attacks," *International Conference on Cryptographic Hardware and Embedded Systems (CHES)*, 2017.

[32] Y. Shen and H. Zhou, "Double DIP: Re-Evaluating Security of Logic Encryption Algorithms," in *Proceedings of the on Great Lakes Symposium on VLSI*, pp. 179–184, 2017.

[33] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, "AppSAT: Approximately deobfuscating integrated circuits," in *Int. Symp. on Hardware Oriented Security and Trust*, 2017.

[34] Y. Shen, A. Rezaei, and H. Zhou, "Sat-based bit-flipping attack on logic encryptions," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 629–632, IEEE, 2018.

[35] D. Sirone and P. Subramanyan, "Functional analysis attacks on logic locking," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 936–939, 2019.

[36] N. Limaye, S. Patnaik, and O. Sinanoglu, "Fa-SAT: Fault-aided SAT-based Attack on Compound Logic Locking Techniques," in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1166–1171, IEEE, 2021.

[37] A. Sengupta, N. Limaye, and O. Sinanoglu, "Breaking cas-lock and its variants by exploiting structural traces," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 418–440, 2021.

[38] Y. Zhong and U. Guin, "Complexity Analysis of the SAT Attack on Logic Locking," *arXiv preprint arXiv:2207.01808*, 2022.

[39] A. Stern, D. Mehta, S. Tajik, U. Guin, F. Farahmandi, and M. Tehranipoor, "SPARTA-COTS: A Laser Probing Approach for Sequential Trojan Detection in COTS Integrated Circuits," in *IEEE Physical Assurance and Inspection of Electronics (PAINE)*, pp. 1–6, 2020.

[40] A. Jain, M. T. Rahman, and U. Guin, "ATPG-Guided Fault Injection Attacks on Logic Locking," in *IEEE Physical Assurance and Inspection of Electronics (PAINE)*, pp. 1–6, 2020.

[41] Y. Zhong and U. Guin, "AFIA: ATPG-Guided Fault Injection Attack on Secure Logic Locking," *arXiv preprint arXiv:2206.04754*, 2022.

[42] Y. Zhong and U. Guin, "Fault-Injection Based Chosen-Plaintext Attacks on Multicycle AES Implementations," in *Proceedings of the Great Lakes Symposium on VLSI 2022*, pp. 443–448, 2022.

[43] A. Jain, Z. Zhou, and U. Guin, "TAAL: Tampering Attack on Any Key-based Logic Locked Circuits," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 26, no. 4, pp. 1–22, 2021.

[44] A. Jain, U. Guin, M. T. Rahman, N. Asadizanjani, D. Duvalsaint, and R. S. Blanton, "Special Session: Novel Attacks on Logic-Locking," in *VLSI Test Symposium (VTS)*, pp. 1–10, 2020.

[45] R. Torrance and D. James, "Reverse engineering in the semiconductor industry," in *IEEE Custom Integrated Circuits Conference*, pp. 429–436, 2007.

[46] R. Torrance and D. James, "The state-of-the-art in semiconductor reverse engineering," in *Proceedings of the Design Automation Conference*, pp. 333–338, 2011.

[47] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, "Security analysis of integrated circuit camouflaging," in *Proceedings of the ACM SIGSAC conference on Computer & communications security*, pp. 709–720, 2013.

[48] B. Erbagci, C. Erbagci, N. E. C. Akkaya, and K. Mai, "A secure camouflaged threshold voltage defined logic family," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 229–235, 2016.

[49] B. Shakya, H. Shen, M. Tehranipoor, and D. Forte, "Covert gates: Protecting integrated circuits with undetectable camouflaging," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 86–118, 2019.

[50] X. Zhang, K. Xiao, and M. Tehranipoor, "Path-delay fingerprinting for identification of recovered ics," in *Proc. International Symposium on Fault and Defect Tolerance in VLSI Systems*, 2012.

[51] K. Huang, J. Carulli, and Y. Makris, "Parametric counterfeit IC detection via Support Vector Machines," in *Proc. International Symposium on Fault and Defect Tolerance in VLSI Systems*, pp. 7–12, 2012.

[52] Y. Zheng, A. Basak, and S. Bhunia, "CACI: Dynamic current analysis towards robust recycled chip identification," in *Design Automation Conference (DAC)*, pp. 1–6, 2014.

[53] H. Dogan, D. Forte, and M. Tehranipoor, "Aging analysis for recycled FPGA detection," in *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, 2014.

[54] Y. Zheng, X. Wang, and S. Bhunia, "SACCI: Scan-Based Characterization Through Clock Phase Sweep for Counterfeit Chip Detection," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2014.

[55] Z. Guo, M. T. Rahman, M. M. Tehranipoor, and D. Forte, "A zero-cost approach to detect recycled SoC chips using embedded SRAM," in *IEEE Int. Symp. on Hardware Oriented Security and Trust*, 2016.

[56] R. Moudgil, D. Ganta, L. Nazhandali, M. Hsiao, C. Wang, and S. Hall, "A novel statistical and circuit-based technique for counterfeit detection in existing ics," in *Proceedings of the 23rd ACM international conference on Great lakes symposium on VLSI*, pp. 1–6, 2013.

[57] G-19A Test Laboratory Standards Development Committee, "AS6171: Test Methods Standard; General Requirements, Suspect/Counterfeit, Electrical, Electronic, and Electromechanical Parts," 2016.

[58] G-19CI Continuous Improvement, "AS5553: Counterfeit Electronic Parts; Avoidance, Detection, Mitigation, and Disposition," 2009.

[59] G-19D Distributor, "AS6081: Fraudulent/Counterfeit Electronic Parts: Avoidance, Detection, Mitigation, and Disposition - Distributors," 2012.

[60] CTI, Certification for Counterfeit Components Avoidance Program, 2011, https://www.cti-us.com/pdf/CCAP101Certification.pdf.

[61] IDEA, Acceptability of Electronic Components Distributed in the Open Market, 2017, http://www.idofea.org/products/118-idea-std-1010b.

[62] C. W. Lin and S. Ghosh, "Novel self-calibrating recycling sensor using schmitt-trigger and voltage boosting for fine-grained detection," in *Sixteenth International Symposium on Quality Electronic Design*, pp. 465–469, IEEE, 2015.

[63] U. Guin, D. Forte, and M. Tehranipoor, "Design of Accurate Low-Cost On-Chip Structures for Protecting Integrated Circuits Against Recycling," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 4, pp. 1233–1246, 2015.

[64] T.-H. Kim, R. Persaud, and C. Kim, "Silicon Odometer: An On-Chip Reliability Monitor for Measuring Frequency Degradation of Digital Circuits," *IEEE Journal of Solid-State Circuits*, pp. 874–880, 2008.

[65] X. Zhang, N. Tuzzio, and M. Tehranipoor, "Identification of recovered ICs using fingerprints from a light-weight on-chip sensor," in *Proc. IEEE-ACM Design Automation Conference*, 2012.

[66] X. Zhang and M. Tehranipoor, "Design of On-Chip Lightweight Sensors for Effective Detection of Recycled ICs," *IEEE Transactions on Very Large Scale Integration Systems*, pp. 1016–1029, 2014.

[67] U. Guin, X. Zhang, D. Forte, and M. Tehranipoor, "Low-Cost On-Chip Structures for Combating Die and IC Recycling," in *ACM/EDAC/IEEE Design Automation Conference (DAC)*, pp. 1–6, 2014.

[68] U. Guin, D. Forte, and M. Tehranipoor, "Design of Accurate Low-Cost On-Chip Structures for Protecting Integrated Circuits Against Recycling," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 1233–1246, 2016.

[69] K. He, X. Huang, and S. X. D. Tan, "EM-based on-chip aging sensor for detection and prevention of counterfeit and recycled ICs," in *IEEE/ACM International Conf. on Computer-Aided Design*, pp. 146–151, 2015.

[70] P. Chowdhury, U. Guin, A. D. Singh, and V. D. Agrawal, "Estimating Operational Age of an Integrated Circuit," *Journal of Electronic Testing*, pp. 1–16, 2021.

[71] W. Wang, U. Guin, and A. Singh, "A Zero-Cost Detection Approach for Recycled ICs using Scan Architecture," in *VLSI Test Symposium (VTS)*, pp. 1–6, 2020.

[72] N. Asadizanjani, M. Tehranipoor, and D. Forte, "Counterfeit electronics detection using image processing and machine learning," in *Journal of Physics: Conference Series*, 2017.

[73] S. Shahbazmohamadi, D. Forte, and M. Tehranipoor, "Advanced physical inspection methods for counterfeit ic detection," in *40th International Symposium for Testing and Failure Analysis*, pp. 55–64, 2014.

[74] M. Miller, J. Meraglia, and J. Hayward, "Traceability in the Age of Globalization: A Proposal for a Marking Protocol to Assure Authenticity of Electronic Parts," in *SAE Aerospace Electronics and Avionics Systems Conference*, 2012.

[75] Semiconductor Industry Association (SIA), "Public Comments - DNA Authentication Marking on Items in FSC5962," 2012.

[76] J. A. Roy, F. Koushanfar, and I. L. Markov, "Ending piracy of integrated circuits," *Computer*, vol. 43, no. 10, pp. 30–38, 2010.

[77] M. Yasin, J. J. Rajendran, O. Sinanoglu, and R. Karri, "On improving the security of logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 9, pp. 1411–1424, 2016.

[78] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *Proc. of ACM/IEEE on Design Automation Conference*, pp. 83–89, 2012.

[79] M. Yasin, B. Mazumdar, J. J. Rajendran, and O. Sinanoglu, "SARLock: SAT attack resistant logic locking," in *Int. Symp. on Hardware Oriented Security and Trust*, pp. 236–241, 2016.

[80] Y. Xie and A. Srivastava, "Anti-sat: Mitigating sat attack on logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 2, pp. 199–207, 2018.

[81] M. Yasin, B. Mazumdar, O. Sinanoglu, and J. Rajendran, "Removal attacks on logic locking and camouflaging techniques," *IEEE Transactions on Emerging Topics in Computing*, 2017.

[82] M. Yasin, B. Mazumdar, O. Sinanoglu, and J. Rajendran, "Security analysis of anti-sat," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 342–347, 2017.

[83] M. Yasin, B. Mazumdar, J. J. Rajendran, and O. Sinanoglu, "Ttlock: Tenacious and traceless logic locking," in *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 166–166, IEEE Computer Society, 2017.

[84] M. Yasin, A. Sengupta, M. Ashraf, M. Nabeel, J. Rajendran, and O. Sinanoglu, "Provably-Secure Logic Locking: From Theory To Practice," in *ACM/SIGSAC Conference on Computer & Communications Security*, pp. 1–1, 2017.

[85] D. Sirone and P. Subramanyan, "Functional analysis attacks on logic locking," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2514–2527, 2020.

[86] C. Pilato, F. Regazzoni, R. Karri, and S. Garg, "Tao: Techniques for algorithm-level obfuscation during high-level synthesis," in *Proceedings of the 55th Annual Design Automation Conference*, pp. 1–6, 2018.

[87] M. R. Muttaki, R. Mohammadivojdan, M. Tehranipoor, and F. Farahmandi, "Hlock: Locking ips at the high-level language," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*, pp. 79–84, IEEE, 2021.

[88] S. E. Quadir, J. Chen, D. Forte, N. Asadizanjani, S. Shahbazmohamadi, L. Wang, J. Chandy, and M. Tehranipoor, "A survey on chip to system reverse engineering," *ACM journal on emerging technologies in computing systems (JETC)*, vol. 13, no. 1, pp. 1–34, 2016.

[89] M. Li, K. Shamsi, T. Meade, Z. Zhao, B. Yu, Y. Jin, and D. Z. Pan, "Provably secure camouflaging strategy for ic protection," *IEEE transactions on computer-aided design of integrated circuits and systems*, vol. 38, no. 8, pp. 1399–1412, 2017.

[90] L.-W. Chow, J. P. Baukus, and W. M. Clark Jr, "Integrated circuits protected against reverse engineering and method for fabricating the same using an apparent metal contact line terminating on field oxide," Nov. 13 2007. US Patent 7,294,935.

[91] M. I. M. Collantes, M. El Massad, and S. Garg, "Threshold-dependent camouflaged cells to secure circuits against reverse engineering attacks," in *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 443–448, IEEE, 2016.

[92] I. R. Nirmala, D. Vontela, S. Ghosh, and A. Iyengar, "A novel threshold voltage defined switch for circuit camouflaging," in *2016 21th IEEE European Test Symposium (ETS)*, pp. 1–2, IEEE, 2016.

[93] S. Patnaik, M. Ashraf, O. Sinanoglu, and J. Knechtel, "Obfuscating the interconnects: Low-cost and resilient full-chip layout camouflaging," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 12, pp. 4466–4481, 2020.

[94] S. Roshanisefat, H. M. Kamali, H. Homayoun, and A. Sasan, "Sat-hard cyclic logic obfuscation for protecting the ip in the manufacturing supply chain," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 4, pp. 954–967, 2020.

[95] M. Yasin, B. Mazumdar, O. Sinanoglu, and J. Rajendran, "Camoperturb: Secure ic camouflaging for minterm protection," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–8, 2016.

[96] J. Zang, M. Mostovoy, J. H. Han, and N. Nagaosa, "Dynamics of skyrmion crystals in metallic thin films," *Physical review letters*, vol. 107, no. 13, p. 136804, 2011.

[97] W. Jiang, X. Zhang, G. Yu, W. Zhang, X. Wang, M. B. Jungfleisch, J. E. Pearson, X. Cheng, O. Heinonen, K. L. Wang, *et al.*, "Direct observation of the skyrmion hall effect," *Nature Physics*, vol. 13, no. 2, pp. 162–169, 2017.

[98] J. Iwasaki, M. Mochizuki, and N. Nagaosa, "Current-induced skyrmion dynamics in constricted geometries," *Nature nanotechnology*, vol. 8, no. 10, pp. 742–747, 2013.

[99] W. Jiang, P. Upadhyaya, W. Zhang, G. Yu, M. B. Jungfleisch, F. Y. Fradin, J. E. Pearson, Y. Tserkovnyak, K. L. Wang, O. Heinonen, *et al.*, "Blowing magnetic skyrmion bubbles," *Science*, vol. 349, no. 6245, pp. 283–286, 2015.

[100] G. Yu, P. Upadhyaya, X. Li, W. Li, S. K. Kim, Y. Fan, K. L. Wong, Y. Tserkovnyak, P. K. Amiri, and K. L. Wang, "Room-temperature creation and spin–orbit torque manipulation of skyrmions in thin films with engineered asymmetry," *Nano letters*, vol. 16, no. 3, pp. 1981–1988, 2016.

[101] X. Zhang, Y. Zhou, M. Ezawa, G. Zhao, and W. Zhao, "Magnetic skyrmion transistor: skyrmion motion in a voltage-gated nanotrack," *Scientific reports*, vol. 5, no. 1, pp. 1–8, 2015.

[102] M. Chauwin, X. Hu, F. Garcia-Sanchez, N. Betrabet, A. Paler, C. Moutafis, and J. S. Friedman, "Skyrmion logic system for large-scale reversible computation," *Physical Review Applied*, vol. 12, no. 6, p. 064053, 2019.

[103] S. Luo, M. Song, X. Li, Y. Zhang, J. Hong, X. Yang, X. Zou, N. Xu, and L. You, "Reconfigurable skyrmion logic gates," *Nano letters*, vol. 18, no. 2, pp. 1180–1184, 2018.

[104] Z. Zhou, U. Guin, P. Li, and V. D. Agrawal, "Fault modeling and test generation for technology-specific defects of skyrmion logic circuits," in *2022 IEEE 40th VLSI Test Symposium (VTS)*, pp. 1–7, IEEE, 2022.

[105] Z. Zhou, U. Guin, P. Li, and V. D. Agrawal, "Defect characterization and testing of skyrmion-based logic circuits," in *2021 IEEE 39th VLSI Test Symposium (VTS)*, pp. 1–7, IEEE, 2021.

[106] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[107] ISO/IEC JTC 1/SC 31, "Information technology automatic identification and data capture techniques QR code bar code symbology specification," *International Organization for Standardization*. ISO/IEC 18004:2015.

[108] D. Bryan, "The ISCAS'85 benchmark circuits and netlist format," *North Carolina State University*, vol. 25, p. 39, 1985.

[109] S. Davidson, "ITC'99 benchmark circuits-preliminary results," in *International Test Conference 1999. Proceedings (IEEE Cat. No. 99CH37034)*, pp. 1125–1125, IEEE Computer Society, 1999.

[110] H. Salmani and M. Tehranipoor, "Trust-hub," 2018. [Online]. Available: https://trust-hub.org/home.

[111] R. Tarjan, "Depth-first search and linear graph algorithms," *SIAM journal on computing*, pp. 146–160, 1972.

[112] Y. Zhang, C. Tang, P. Li, and U. Guin, "CamSkyGate: Camouflaged Skyrmion Gates for Protecting ICs," in *Design Automation Conference (DAC)*, pp. 1–6, 2022.

[113] ECID vs Device ID, Bill Eklow, 2006, `btw.tttc-events.org/material/BTW10/Presentations/Session%205.2.pptx`.

[114] IEEE 1149.1-2013 - IEEE Standard for Test Access Port and Boundary-Scan Architecture, `https://standards.ieee.org/standard/1149_1-2013.html`.

[115] T.-K. Lee, "Process monitor for CMOS integrated circuits," Jan. 23 1996. US Patent 5,486,786.

[116] E. O. Sugasawara, "Process monitor circuitry for integrated circuits," 2000. US Patent 6,124,143.

[117] R. Bach, "Process monitor with statistically selected ring oscillator," 2003. US Patent 6,544,807.

[118] National Institute of Standards and Technology, "FIPS 180-4: Secure Hash Standard (SHS)," 2015.

[119] National Institute of Standards and Technology, "FIPS 186-4: Digital Signature Standard (DSS)," 2013.

[120] K. Bernstein, "Supply chain hardware integrity for electronics defense (SHIELD)," *Defense Advanced Research Projects Agency, Microsystems Technology Office/MTO Broad Agency Announcement*, 2014.

[121] D. R. Kuhn, V. C. Hu, W. T. Polk, and S.-J. Chang, "Introduction to public key technology and the federal PKI infrastructure," tech. rep., National Inst of Standards and Technology, 2001.

[122] NXP, MIFARE Classic EV1 1K - Mainstream contactless smart card IC for fast and easy solution development, 2018, `https://www.nxp.com/docs/en/data-sheet/MF1S50YYX_V1.pdf`.

[123] NXP, MFRC522: Standard performance MIFARE and NTAG frontend, 2016, `https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf`.

[124] M2Crypto, `https://gitlab.com/m2crypto/m2crypto`.

[125] ECDSA_AllPrime-Elliptic Curve Digital Signature Algorithm, `https://csrc.nist.gov/csrc/media/projects/cryptographic-standards-and-guidelines/documents/examples/ecdsa_prime.pdf`.

[126] C. Helfmeier, D. Nedospasov, C. Tarnovsky, J. S. Krissler, C. Boit, and J.-P. Seifert, "Breaking and entering through the silicon," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pp. 733–744, 2013.

[127] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[128] L. Cheng, F. Liu, and D. Yao, "Enterprise data breach: causes, challenges, prevention, and future directions," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 7, no. 5, p. e1211, 2017.

[129] C. Cadwalladr and E. Graham-Harrison, "Revealed: 50 million facebook profiles harvested for cambridge analytica in major data breach," *The guardian*, vol. 17, p. 22, 2018.

[130] Y. Zhong and U. Guin, "Chosen-Plaintext Attack on Energy-Efficient Hardware Implementation of GIFT-COFB," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 1–4, 2022.

[131] E. Castillo, U. Meyer-Baese, A. García, L. Parrilla, and A. Lloris, "IPP@HDL: Efficient Intellectual Property Protection Scheme for IP Cores," *IEEE Trans. Very Large Scale Integr. Syst.*, pp. 578–591, 2007.

[132] R. Chakraborty and S. Bhunia, "Hardware protection and authentication through netlist level obfuscation," in *Proc. of IEEE/ACM International Conference on Computer-Aided Design*, pp. 674 –677, 2008.

[133] E. Charbon, "Hierarchical watermarking in IC design," in *Custom Integrated Circuits Conference*, pp. 295–298, 1998.

[134] A. Kahng, J. Lach, W. Mangione-Smith, S. Mantik, I. Markov, M. Potkonjak, P. Tucker, H. Wang, and G. Wolfe, "Constraint-based watermarking techniques for design IP protection," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1236–1252, 2001.

[135] G. Qu and M. Potkonjak, *Intellectual property protection in VLSI designs: theory and practice*. Springer Science & Business Media, 2003.

[136] R. W. Jarvis and M. G. McIntyre, "Split manufacturing method for advanced semiconductor circuits," 2007. US Patent 7,195,931.

[137] K. Vaidyanathan, R. Liu, E. Sumbul, Q. Zhu, F. Franchetti, and L. Pileggi, "Efficient and secure intellectual property (IP) design with split fabrication," in *Int. Symp. on Hardware Oriented Security and Trust*, pp. 13–18, 2014.

[138] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *International Symposium on Hardware Oriented Security and Trust*, pp. 137–143, 2015.

[139] L. Alrahis, M. Yasin, N. Limaye, H. Saleh, B. Mohammad, M. Alqutayri, and O. Sinanoglu, "ScanSAT: Unlocking Static and Dynamic Scan Obfuscation," *Transactions on Emerging Topics in Computing*, 2019.

[140] K. Shamsi, M. Li, K. Plaks, S. Fazzari, D. Z. Pan, and Y. Jin, "IP Protection and Supply Chain Security through Logic Obfuscation: A Systematic Overview," *Trans. on Design Automation of Electronic Systems (TODAES)*, p. 65, 2019.

[141] X. Wang, D. Zhang, M. He, D. Su, and M. Tehranipoor, "Secure scan and test using obfuscation throughout supply chain," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 9, pp. 1867–1880, 2017.

[142] A. Sengupta, M. Nabeel, N. Limaye, M. Ashraf, and O. Sinanoglu, "Truly stripping functionality for logic locking: A fault-based perspective," *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020.

[143] R. Karmakar, S. Chatopadhyay, and R. Kapur, "Encrypt flip-flop: A novel logic encryption technique for sequential circuits," *arXiv preprint arXiv:1801.04961*, 2018.

[144] S. Potluri, A. Aysu, and A. Kumar, "Seql: Secure scan-locking for ip protection," *arXiv preprint arXiv:2005.13032*, 2020.

[145] H.-Y. Chiang, Y.-C. Chen, D.-X. Ji, X.-M. Yang, C.-C. Lin, and C.-Y. Wang, "LOOPLock: LOgic OPtimization based Cyclic Logic Locking," *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2019.

[146] K. Juretus and I. Savidis, "Increasing the SAT Attack Resiliency of In-Cone Logic Locking," in *International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2019.

[147] B. Shakya, X. Xu, M. Tehranipoor, and D. Forte, "Cas-lock: A security-corruptibility trade-off resilient logic locking scheme," *IACR Transactions on Cryptographic Hardware and Embedded Systems*, pp. 175–202, 2020.

[148] N. Limaye, A. Sengupta, M. Nabeel, and O. Sinanoglu, "Is Robust Design-for-Security Robust Enough? Attack on Locked Circuits with Restricted Scan Chain Access," *arXiv preprint arXiv:1906.07806*, 2019.

[149] K. Shamsi, M. Li, D. Z. Pan, and Y. Jin, "Kc2: Key-condition crunching for fast sequential circuit deobfuscation," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 534–539, IEEE, 2019.

[150] B. Shakya, X. Xu, M. Tehranipoor, and D. Forte, "Defeating cas-unlock.," *IACR Cryptol. ePrint Arch.*, vol. 2020, p. 324, 2020.

[151] H. Chen, C. Fu, J. Zhao, and F. Koushanfar, "Genunlock: An automated genetic algorithm framework for unlocking logic encryption.," in *ICCAD*, pp. 1–8, 2019.

[152] Q. Tan, S. Potluri, and A. Aysu, "Efficacy of satisfiability based attacks in the presence of circuit reverse engineering errors," *arXiv preprint arXiv:2005.13048*, 2020.

[153] UF/FICS Hardware De-obfuscation Competition (2019) Available: https://trust-hub.org/competitions/hwobfuscation1.

[154] Trusted and Assured Micro Electronics Forum (2019). Available: `https://www.tameforum.org/`.

[155] A. J. Reich, K. H. Nakagawa, and R. E. Boone, "OASIS vs. GDSII stream format efficiency," in *23rd Annual BACUS Symposium on Photomask Technology*, vol. 5256, pp. 163–174, 2003.

[156] R. Torrance and D. James, "The state-of-the-art in IC reverse engineering," in *International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 363–381, 2009.

[157] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2009.

[158] Python-2.7, Available: https://www.python.org/download/releases/2.7/(2019).

[159] P. J. Dickinson, H. Bunke, A. Dadej, and M. Kraetzl, "On graphs with unique node labels," in *International Workshop on Graph-Based Representations in Pattern Recognition*, pp. 13–23, Springer, 2003.

[160] A. Abboud, A. Backurs, T. D. Hansen, V. Vassilevska Williams, and O. Zamir, "Subtree isomorphism revisited," *ACM Transactions on Algorithms (TALG)*, vol. 14, no. 3, p. 27, 2018.

[161] UF/FICS Hardware De-obfuscation Competition, 2019, https://trust-hub.org/competitions/hwobfuscation1.

[162] HeLLO: CTF '21, Attacks on Hardware Logic Locking & Obfuscation Capture The Flag, 2021, https://helloctf.org/21/winners/.

[163] T. Chen, "Overcoming research challenges for cmos scaling: Industry directions," in *International Conference on Solid-State and Integrated Circuit Technology Proceedings*, pp. 4–7, 2006.

[164] A. Fert, V. Cros, and J. Sampaio, "Skyrmions on the track," *Nature nanotechnology*, vol. 8, no. 3, pp. 152–156, 2013.

[165] Z. Zhou, U. Guin, P. Li, and V. D. Agrawal, "Fault Modeling and Test Generation for Technology-Specific Defects of Skyrmion Logic Circuits," in *VLSI Test Symposium (VTS)*, pp. 1–7, 2022.

[166] C. Tang, L. Alahmed, J. Xu, M. Shen, N. A. Jones, M. Sadi, U. Guin, W. Zhao, and P. Li, "Effects of Temperature and Structural Geometries on a Skyrmion Logic Gate," *IEEE Transactions on Electron Devices*, pp. 1706–1712, 2021.

[167] M. Sadi, P. Li, U. Guin, S. Walters, and D. DiMase, "Low Power, Rad-Hard, and Secure Polymorphic and Neuromorphic Designs using Skyrmions," in *GOMACTech*, pp. 1–4, 2022.

[168] U. Guin, D. DiMase, and M. Tehranipoor, "A Comprehensive Framework for Counterfeit Defect Coverage Analysis and Detection Assessment," *Journal of Electronic Testing*, vol. 30, no. 1, pp. 25–40, 2014.

[169] W. Kang, Y. Huang, X. Zhang, Y. Zhou, and W. Zhao, "Skyrmion-electronics: An overview and outlook," *Proceedings of the IEEE*, vol. 104, no. 10, pp. 2040–2061, 2016.

[170] S. Luo and L. You, "Skyrmion devices for memory and logic applications," *APL Materials*, vol. 9, no. 5, p. 050901, 2021.

[171] D. M. Crum, M. Bouhassoune, J. Bouaziz, B. Schweflinghaus, S. Blügel, and S. Lounis, "Perpendicular reading of single confined magnetic skyrmions," *Nature Communications*, vol. 6, no. 1, 2015.

[172] B. W. Walker, C. Cui, F. Garcia-Sanchez, J. A. C. Incorvia, X. Hu, and J. S. Friedman, "Skyrmion logic clocked via voltage-controlled magnetic anisotropy," *Applied Physics Letters*, vol. 118, no. 19, p. 192404, 2021.

[173] A. Thiele, "Steady-state motion of magnetic domains," *Physical Review Letters*, vol. 30, no. 6, p. 230, 1973.

[174] S. Breitkreutz, G. Ziemys, I. Eichwald, J. Kiermaier, G. Csaba, W. Porod, D. Schmitt-Landsiedel, and M. Becherer, "Domain wall gate for magnetic logic and memory applications with perpendicular anisotropy," in *IEEE International Electron Devices Meeting*, pp. 22–4, 2013.

[175] J. McCord, I. Mönch, J. Fassbender, A. Gerber, and E. Quandt, "Local setting of magnetic anisotropy in amorphous films by co ion implantation," *Journal of Physics D: Applied Physics*, vol. 42, no. 5, p. 055006, 2009.

[176] M. Gavagnin, H. D. Wanzenboeck, S. Wachter, M. M. Shawrav, A. Persson, K. Gunnarsson, P. Svedlindh, M. Stoger-Pollach, and E. Bertagnolli, "Free-standing magnetic nanopillars for

3d nanomagnet logic," *ACS applied materials & interfaces*, vol. 6, no. 22, pp. 20254–20260, 2014.

[177] N. Gaur, S. Kundu, S. Piramanayagam, S. Maurer, H. Tan, S. Wong, S. Steen, H. Yang, and C. Bhatia, "Lateral displacement induced disorder in l1 0-fept nanostructures by ion-implantation," *Scientific reports*, vol. 3, no. 1, pp. 1–7, 2013.

[178] S. Y. Park, D. S. Kim, Y. Liu, J. Hwang, *et al.*, "Controlling the magnetic anisotropy of the van der waals ferromagnet fe3gete2 through hole doping," *Nano Letters*, vol. 20, no. 1, pp. 95–100, 2020.

[179] L. Frank, M. Hovorka, M. El-Gomati, I. Müllerová, F. Mika, and E. Mikmeková, "Acquisition of the dopant contrast in semiconductors with slow electrons," *Journal of Electron Spectroscopy and Related Phenomena*, vol. 241, p. 146836, 2020.

[180] C. S. Simulations. `https://github.com/2660039863/CamSkyGate_DAC`, 2021.

[181] M. El Massad, S. Garg, and M. V. Tripunitara, "Integrated circuit (ic) decamouflaging: Reverse engineering camouflaged ics within minutes.," in *NDSS*, pp. 1–14, 2015.

[182] L. Amarú, P. Gaillardon, and G. D. Micheli, "The epfl combinational benchmark suite," in *Proceedings of the International Workshop on Logic & Synthesis (IWLS)*, no. CONF, 2015.

[183] Y. Xie and A. Srivastava, "Mitigating sat attack on logic locking," in *International conference on cryptographic hardware and embedded systems*, pp. 127–146, 2016.

[184] Y. Zhang and U. Guin, "End-to-End Traceability of ICs in Component Supply Chain for Fighting Against Recycling," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 767–775, 2019.

[185] U. Guin, W. Wang, C. Harper, and A. D. Singh, "Detecting Recycled SOCs by Exploiting Aging Induced Biases in Memory Cells," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 72–80, 2019.

[186] P. Chowdhury, U. Guin, A. D. Singh, and V. D. Agrawal, "Two-pattern Δ_{IDDQ} Test for Recycled IC Detection," in *International Conference on VLSI Design and International Conference on Embedded Systems (VLSID)*, pp. 82–87, 2019.

[187] M. Alam, S. Chowdhury, M. M. Tehranipoor, and U. Guin, "Robust, Low-Cost, and Accurate Detection of Recycled ICs using Digital Signatures," in *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pp. 209–214, 2018.

[188] B. Shakya, U. Guin, M. Tehranipoor, and D. Forte, "Performance Optimization for On-Chip Sensors to Detect Recycled ICs," in *IEEE International Conference on Computer Design (ICCD)*, pp. 289–295, 2015.

[189] U. Guin, D. Forte, and M. Tehranipoor, " Low-cost On-Chip Structures for Combating Die and IC Recycling ," *GOMACTech*, 2014.

[190] P. Cui, J. Dixon, U. Guin, and D. DiMase, "A blockchain-based framework for supply chain provenance," *IEEE Access*, vol. 7, pp. 157113–157125, 2019.

[191] P. Cui, U. Guin, A. Skjellum, and D. Umphress, "Blockchain in IoT: Current Trends, Challenges, and Future Roadmap," *Journal of Hardware and Systems Security*, vol. 3, no. 4, pp. 338–364, 2019.

[192] P. Cui and U. Guin, "Countering Botnet of Things using Blockchain-Based Authenticity Framework," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 598–603, 2019.

[193] U. Guin, P. Cui, and A. Skjellum, "Ensuring Proof-of-Authenticity of IoT Edge Devices using Blockchain Technology," in *IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pp. 1042–1049, 2018.

[194] M. N. Islam, V. C. Patii, and S. Kundu, "On ic traceability via blockchain," in *2018 International symposium on VLSI design, automation and test (VLSI-DAT)*, pp. 1–4, IEEE, 2018.

[195] X. Xu, F. Rahman, B. Shakya, A. Vassilev, D. Forte, and M. Tehranipoor, "Electronics supply chain integrity enabled by blockchain," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 24, no. 3, pp. 1–25, 2019.

[196] N. Vashistha, M. M. Hossain, M. R. Shahriar, F. Farahmandi, F. Rahman, and M. M. Tehranipoor, "echain: A blockchain-enabled ecosystem for electronic device authenticity verification," *IEEE Transactions on Consumer Electronics*, vol. 68, no. 1, pp. 23–37, 2021.

[197] M. N. Islam and S. Kundu, "Enabling ic traceability via blockchain pegged to embedded puf," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 24, no. 3, pp. 1–23, 2019.

[198] A. Pouraghily, M. N. Islam, S. Kundu, and T. Wolf, "Privacy in blockchain-enabled iot devices," in *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)*, pp. 292–293, IEEE, 2018.