AFRL-RY-WP-TR-2021-0204

MEMOMETER: STRONG PUF-BASED PASSIVE MEMORY HARDWARE METERING METHODOLOGY FOR INTEGRATED CIRCUITS (Preprint)

Anvesh K. Perumalla
University of Cincinnati

AUGUST 2021
Final Report

AIR FORCE RESEARCH LABORATORY
SENSORS DIRECTORATE
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7320
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
August 2021	Dissertation	13 July 2021 –13 July 2021

4. TITLE AND SUBTITLE
MEMOMETER: STRONG PUF-BASED PASSIVE MEMORY HARDWARE METERING METHODOLOGY FOR INTEGRATED CIRCUITS (Preprint)

5a. CONTRACT NUMBER
FA8650-14-D-1724-0003

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62204F

6. AUTHOR(S)
Anvesh K. Perumalla

5d. PROJECT NUMBER
2002

5e. TASK NUMBER
N/A

5f. WORK UNIT NUMBER
Y18S

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Cincinnati
2600 Clifton Ave.
Cincinnati, OH 45221

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory
Sensors Directorate
Wright-Patterson Air Force Base, OH 45433-7320
Air Force Materiel Command
United States Air Force

10. SPONSORING/MONITORING AGENCY ACRONYM(S)
AFRL/RYDT

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-RY-WP-TR-2021-0204

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. Approved for public release. Distribution is unlimited.

13. SUPPLEMENTARY NOTES
PAO case number AFRL-2021-2224, Clearance Date 13 July 2021. Submitted to the Graduate School of the University of Cincinnati in partial fulfillment of the requirement for the degree of Doctor of Philosophy in the Department of Electrical Engineering and Computer Science of the College of Engineering and Applied Science. Report contains color.

14. ABSTRACT

As the semiconductor business has shifted towards a “fabless” model (horizontal business model); integrated circuit (IC) supply chain has become untrustworthy. ICs have altered the way we live, function, work, etc. To keep up with the demand, and for economic incentives, ICs are designed at one place and the fabrication happens at another place. This business model is beneficial for the sake of revenue but involves compromised trust, since the fabrication contains all the design files necessary to overbuild, counterfeit, and clone ICs. Hence the identification and authentication of ICs has become an enormous concern for the government and commercial entities. In order to overcome these challenges, we present the Memometer: a low-cost, low-overhead system which maintains control and accountability of field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), and programmable logic devices (PLDs). The Memometer is based on passive hardware metering methodology that can uniquely tag each IC with an unclonable fingerprint. These fingerprints are further used in identification and authentication within the supply chain. The Memometer system uses uncommitted memory start-up values (SUVs) to create memory signatures and to identify and authenticate any IC using these signatures. The Memometer uses the IEEE 1149.x Join Test Action Group (JTAG) port, readily available on almost all ICs, to fingerprint ICs. ……continued on next page

15. SUBJECT TERMS
aging analysis, counterfeit electronics, fingerprinting, hardware metering, IC cloning, PUF

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT:	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON (Monitor)
a. REPORT	b. ABSTRACT	c. THIS PAGE			Vipul Patel
Unclassified	Unclassified	Unclassified	SAR	106	19b. TELEPHONE NUMBER *(Include Area Code)* N/A

A generic hardware descriptive language (HDL) kernel is used to read power-on SUVs of unconfigured SRAM and D-FF memory cells by accessing the JTAG port available on ICs. A strong physically unclonable function (PUF) statistical model is used to create and authenticate fingerprints based on these memory SUVs. One of the issues in applying memory PUFs to FPGAs is that contemporary FPGAs come with initialized power-on memory SUVs. This defeats the purpose of applying memory PUFs to these newer FPGAs. To overcome this manufacturing preset, we have developed our own memory PUF approach based on cross-coupled FPGA look-up tables (LUTs) that form non-SUV memory cells that are not preset at power-on. We have demonstrated our methodology on ten Xilinx FPGAs. Our results show that these SUVs are unique and reproducible with an average inter-chip hamming distance (HD) of 49.68% to an ideal of 50% and an average intra-chip HD of 0.87% to an ideal of 0% for a 64-bit fingerprint. These unique, unclonable SUVs are further used to create and authenticate digital fingerprints. Instead of having one fingerprint per device, our methodology makes provisions for many (hundreds) of fingerprints per device, making it a strong PUF. We also investigate and analyze ageing artifacts of our fingerprints using accelerated ageing techniques to predict the reliability of the fingerprints over a five-year period.

Memometer: Strong PUF-Based Passive Memory Hardware Metering Methodology for Integrated Circuits

A dissertation submitted to the Graduate School of the University of Cincinnati in partial fulfillment of the requirement for the degree of

Doctor of Philosophy

in the Department of Electrical Engineering and Computer Science of the College of Engineering and Applied Science

by

Anvesh K. Perumalla
B.E., Osmania University, India, 2012
M.S., Wright State University, U.S.A, 2016

Committee Chair: John (Marty) Emmert, Ph.D.

June 2021

Abstract

As the semiconductor business has shifted towards a “fabless” model (horizontal business model); integrated circuit (IC) supply chain has become untrustworthy. ICs have altered the way we live, function, work, etc. To keep up with the demand, and for economic incentives, ICs are designed at one place and the fabrication happens at another place. This business model is beneficial for the sake of revenue but involves compromised trust, since the fabrication contains all the design files necessary to overbuild, counterfeit, and clone ICs. Hence the identification and authentication of ICs has become an enormous concern for the government and commercial entities.

In order to overcome these challenges, we present the Memometer: a low-cost, low-overhead system which maintains control and accountability of field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), and programmable logic devices (PLDs). The Memometer is based on passive hardware metering methodology that can uniquely tag each IC with an unclonable fingerprint. These fingerprints are further used in identification and authentication within the supply chain.

The Memometer system uses uncommitted memory start-up values (SUVs) to create memory signatures and to identify and authenticate any IC using these signatures. The Memometer uses the IEEE 1149.x Join Test Action Group (JTAG) port, readily available on almost all ICs, to fingerprint ICs. A generic hardware descriptive language

(HDL) kernel is used to read power-on SUVs of unconfigured SRAM and D-FF memory cells by accessing the JTAG port available on ICs. A strong physically unclonable function (PUF) statistical model is used to create and authenticate fingerprints based on these memory SUVs. One of the issues in applying memory PUFs to FPGAs is that contemporary FPGAs come with initialized power-on memory SUVs. This defeats the purpose of applying memory PUFs to these newer FPGAs.

To overcome this manufacturing preset, we have developed our own memory PUF approach based on cross-coupled FPGA look-up tables (LUTs) that form non-SUV memory cells that are not preset at power-on. We have demonstrated our methodology on ten Xilinx FPGAs. Our results show that these SUVs are unique and reproducible with an average inter-chip hamming distance (HD) of 49.68% to an ideal of 50% and an average intra-chip HD of 0.87% to an ideal of 0% for a 64-bit fingerprint. These unique, unclonable SUVs are further used to create and authenticate digital fingerprints. Instead of having one fingerprint per device, our methodology makes provisions for many (hundreds) of fingerprints per device, making it a strong PUF. We also investigate and analyze ageing artifacts of our fingerprints using accelerated ageing techniques to predict the reliability of the fingerprints over a five-year period.

Keywords *– Metering*, *JTAG*, *physically unclonable functions*, *memory PUF*, *strong PUF*, *FPGAs*, *ASICs, PLDs, fingerprinting, IC counterfeiting, IC cloning, aging analysis.*

Acknowledgements

This research was funded by the United States Air Force through Edaptive Computing Inc. under Contract FA8650-14-D-1724-0003.

Dedication

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1 Motivation / Problem

Integrated Circuits (ICs) are the "heart" of every electronic system. These systems are used in healthcare, transportation, finances, communication, aerospace, electric power grids, the military, and beyond. Trust and assurance of these systems are vitally important. Due to the recent shift in the semiconductor business model to "fabless," the IC supply chain has become more vulnerable to the attacks. This has created an avenue for intellectual property (IP) theft, overbuilding, counterfeiting, and cloning of ICs.

1.1.1 Semiconductor Supply Chain Security

The IC life cycle starts from acquiring IP from a third party to be integrated in to the designs, to generating the layouts (GDS II or OASIS format) to be sent to the foundry, to testing the ICs in the third party test facility, to integrating these ICs into the supply chain environment [41]. Malicious activity can occur at any stage during the IC life cycle [12]. The International Chamber of Commerce predicts that by 2022 there will be at least $2 trillion of counterfeit products and a loss of up to 5.4 million jobs globally [3]. Here are a few reports of the hardware vulnerabilities found due to the untrusted IC life cycle or the lack of implanting secure chip design methodologies:

- General Patrick O'Reilly, director of Missile Defense Agency stated that "we do not want a $12 million missile defense interceptor's reliability compromised by a $2 counterfeit part" [44]. The Department of Defense (DOD) supply chain inquiry report from 2012 shows that, just in 2009 and 2010, they have uncovered 1,800

cases of counterfeit electronic parts, with a total number of over a million suspect parts from those cases. These parts would have gone into secure defense systems.

- Counterfeit Intel and Xilinx chips were sold to US military contractors [33]. If a malicious agent replaces a genuine IC with the counterfeit IC in the inventory, do we have a methodology in place to identify the counterfeit IC before it goes into the mission critical system?
- Another incident shows that U.S. customs seized more than five million counterfeit ICs designated for military contractors and commercial aviation [9].
- A tiny microchip which was not part of original motherboard's design, was discovered in DOD data centers, the CIA's drone operations, and onboard networks of Navy warships. This microchip acted as a secret doorway to retrieve sensitive private information [2].
- The Meltdown and Spectre attacks exploited the vulnerabilities in computer processors. These vulnerabilities were found in the computer chips that were made in the last two decades. Reports showed that a malicious attacker could steal sensitive data stored in the protected memory by launching these attacks [28,34].
- A MOLES (Malicious Off-chip Leakage Enabled by Side-channels) circuit is a small hidden circuit (less than 50 gates) implanted in an Advanced Encryption Standard (AES) cryptographic module as shown in Figure 1.1 [33]. By using a simple signal processing technique, a malicious agent can monitor the power spikes to extract the cryptographic secret key. This MOLES circuit acts as a backdoor to the genuine designs which were implanted by the untrusted foundry

without the knowledge of the IP owner. Such a small trojan circuit often goes undetected using standard chip testing.

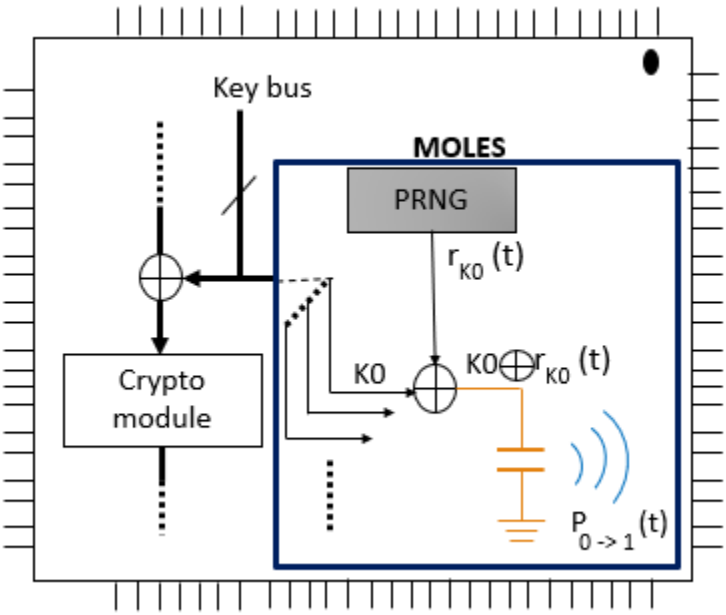


Figure 1. 1 MOLES circuit embedded into a crypto processor [33]

- Another recent FPGA attack called "Starbleed" discovered the vulnerability in the state-of-the-art Xilinx FPGAs that exist in the market today [3, 9]. Using this vulnerability, researchers have exploited a design flaw in the Xilinx 7-series FPGA. This design flaw makes provision for leaking a decrypted FPGA bitstream. Researchers not only showed that they can decrypt a supposedly secure bitstream but can also maliciously manipulate it using a very low-cost tool.

In addition to counterfeit electronics, overbuilding ICs has become a huge concern for government and commercial industrialists. Semiconductor companies have been gradually shifting to the horizontal business model. In a horizontal business model, also known as a "fabless" model, designs are conducted in a trusted facility, and fabrication is conducted in an untrusted facility [32]. As mentioned earlier, since the

fabrication facility has access to all the design files, they can easily overproduce ICs without the designer's knowledge.

A recent McKinsey report on semiconductors [26] shows that the "fabless" semiconductor business model captures the most shareholder value of all the other business models, due to the demanding needs of smartphones and tablets. A 2020 State of the U.S. semiconductor industry report by the Semiconductor Industry Association (SIA) [1] stated that the "United States today now only accounts for 12.5 percent of total installed semiconductor manufacturing, with more than 80 percent of production now happening in Asia." This SIA report also shows that state-of-the-art 7-nm and below IC production is happening exclusively outside of the United States. This creates an opportunity for untrusted agents to overproduce ICs and place them in the supply chain.

Figure 1.2 shows the IC overbuilding threat model where the untrusted foundry - since it has access to all the design files to manufacture an IC - might fabricate more ICs and place them in the supply chain along with the genuine ICs [32]. The untrusted foundry, of course, does not report these overproduced ICs. When these extra ICs are placed in the supply chain along with the genuine ICs, the market value of these genuine ICs drops suddenly, causing a catastrophe for the IP rights owner.

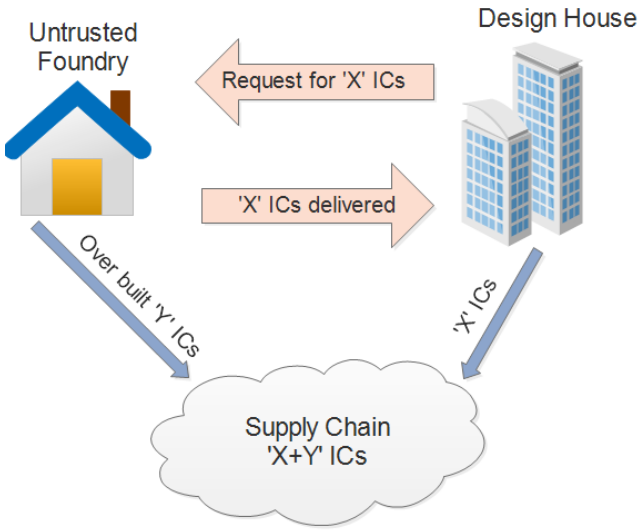


Figure 1. 2 Threat model for IC overbuilding

For example, as shown in Figure 1.2, the design house requests ‘X’ ICs from the foundry. Since the foundry has all the design files to manufacture the ICs, they make ‘X+Y’ ICs and only deliver ‘X’ ICs to the design house. With a total of ‘X+Y’ ICs in the supply chain, now there are ‘Y’ ICs that are not genuine. Thus, there is a problem with the overbuilding of ICs without the knowledge of the design house. Hardware metering helps in identify genuine ‘X’ ICs from the overbuilt ‘Y’ ICs within the supply chain. Passive metering helps in identifying the ICs whereas active metering helps not only in identifying but also locking and unlocking the IC functionality by the design house.

1.1.2 Internet of Things (IoT) Devices

As we know, ICs have radically transformed our lifestyle. Transistor density has tracked with Moore’s prediction for the last few decades; ICs have been manufactured for extremely low prices with higher computation capabilities. Embedded systems and Internet of Things (IoT) have played a major role in this explosion of devices. A white

paper from Cisco in 2011 predicted that there would be at least 50 billion connected devices by 2020 [10]. Peter Hartwell from HP Labs said, “With a trillion sensors embedded in the environment – all connected by computing systems, software, and services – it will be possible to hear the heartbeat of the Earth, impacting human interaction with the globe as profoundly as the internet has revolutionized communication” [10]. The IoT revolution has made this statement believable. IoT devices are basically embedded into every major sector that we can think of: transportation, finances, banking, communication, etc. A threat model for counterfeit IoT devices is shown in Figure 1.3. Malicious IoT devices are implanted into the supply chain by an untrusted original equipment manufacturer (OEM). These malicious devices include counterfeit, cloned, and pirated ICs. The problem exists when we cannot identify the genuine devices from the pirated ones.

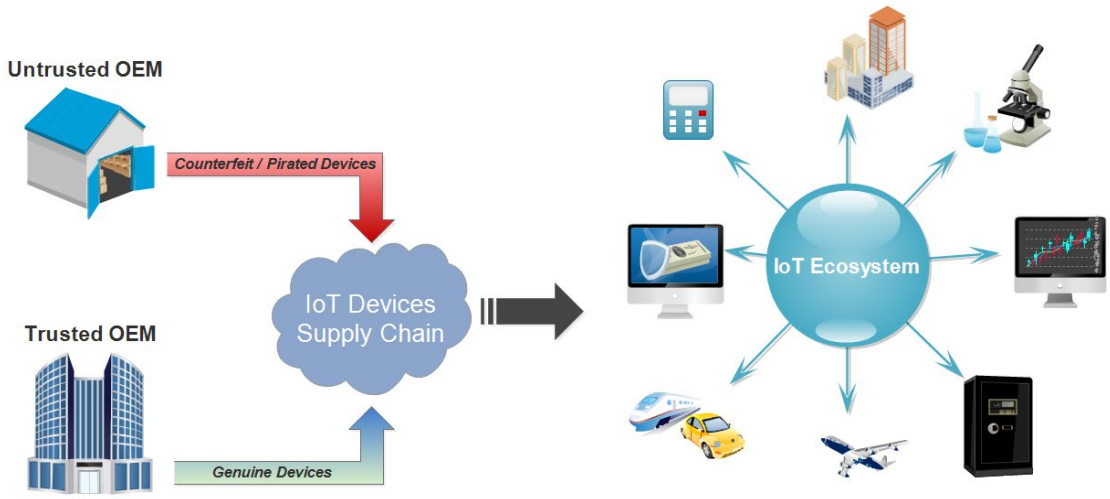


Figure 1. 3 Threat model for IoT counterfeit devices

1.1.3 Defense Mechanisms

Here is a brief overview of a few state-of-the art defense mechanisms to combat against IC overbuilding, counterfeiting, and IP piracy [47].

1.1.3.1 Watermarking

Digital watermarking is a well-established field in multimedia and data protection [42]. The basic idea of applying the watermarking concept to securing ICs is that the designer embeds a unique signature in an IP, often implemented during the physical design phase of IC design [27]. This watermark can be later used to claim ownership of that IC. Watermarking is a technique used to identify the design, whereas fingerprinting/metering uniquely identifies the IC manufactured under the same mask [31]. Watermarking, just by itself, is not an effective hardware security technique unless it is paired with other security measures.

1.1.3.2 Fingerprinting

Cryptographic keys are often stored in non-volatile memory (NVM) or external volatile memory using a battery backup. This traditional way of embedding crypto keys into the IC adds overhead, becomes expensive, and can often become complex [45]. Hence, this has led to a new security approach based on physically unclonable functions (PUFs). Silicon PUFs basically create hard-to-forge, unique fingerprints based on internal manufacturing variations of an IC [53]. These fingerprints are further used to secure ICs against overbuilding, counterfeiting, and piracy. The Memometer uses a new type of PUF methodology to secure reconfigurable ICs. More details of this methodology are found in chapter 3.

1.1.3.3 Obfuscation

IC obfuscation is a technique used to hide the details of the design by adding key-based additional gates into the design [48, 44, 57]. The correct key sequence is needed to unlock the design. A simple example of an obfuscated circuit is shown in Figure 1.4, where K1 and K2 are the key bits. In order for the design to function properly, the key bits (K1,K2) = (0,1) should be used. Otherwise, the design will output incorrectly.

Malicious parties may be able to decipher the IC by using various pattern recognition techniques. For example, they can observe different I/O patterns and propagate the key bit to the output [44]. Once the IC/IP is unlocked then they may be able to create counterfeits and sell them in the black market for a cheaper price.

Securing ICs through obfuscation is beyond the scope of our work. Nevertheless, our metering methodology could be used for IC obfuscation. IC obfuscation is an ongoing research area in the field of hardware security.

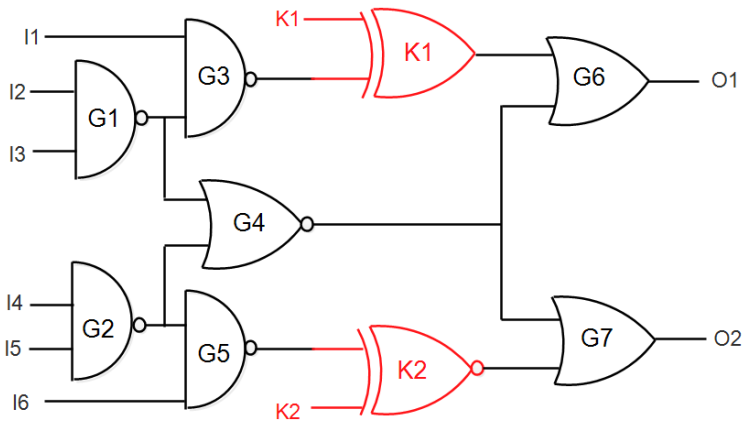


Figure 1. 4 A simple obfuscation technique [44]

1.1.3.4 Split Manufacturing

Split manufacturing is a type of obfuscation technique where IC front-end of the line (FEOL) is fabricated at an untrusted location and back-end-of the line (BEOL) is fabricated at another trusted location [43]. This methodology obfuscates the design details from the untrusted foundry. Without knowing the top metal layer routing details, it is impossible to know the IC's functionality. In order to counterfeit or pirate an IC fabricated using split manufacturing, malicious attackers need to reverse engineer the IC, which is a tedious and expensive process. Split manufacturing combined with asynchronous design is even more challenging for reverse engineering and side-channel attacks [18, 7, 8].

1.1.3.5 IC Aging Detection

Determining an IC's age can be an effective technique to combat against counterfeit/recycled ICs [15]. An IC's lifetime is influenced by negative bias temperature instability (NBTI), hot carrier injection, and electromagnetic migration [47, 58]. Measurements can be taken from trusted ICs using artificial aging techniques. These measurements can be used to identify whether an IC is new or recycled.

1.1.3.6 Metering

Metering helps to gain authorship of an IC/IP after fabrication by uniquely locking/tagging each IC that is manufactured under the same mask [32]. Hardware metering is classified as passive and active [31]. The passive metering technique is used to tag each IC with an unclonable unique identifier. This identifier is further used in identifying genuine ICs from the overbuilt/counterfeit ICs. Active metering helps in preventing the ICs from entering into the supply chain. A more detailed study on

metering is described in the background study section. The Memometer is a passive memory-based metering methodology. In the next section, we give a brief overview of the Memometer.

1.2 Memometer

Memory based physically unclonable functions (PUFs) offer a low cost and low overhead solution for IC identification and protection against counterfeiting and cloning. A memory PUF is created by taking advantage of the memory start-up values (SUVs) to create unique unclonable fingerprints. When this memory PUF technology is paired with hardware metering, it helps to create a robust identification / authentication of the ICs.

The Memometer system is *memo*-ry PUF based hardware *meter*-ing methodology used to maintain control and accountability of application specific integrated circuits (ASICs), field programmable gate arrays (FPGAs), and programmable logic devices (PLDs). The Memometer is a low-cost, low-overhead, adaptable, metering system, which works for any IC that adheres to the IEEE 1149.x Joint Test Action Group (JTAG) standard. The Memometer system uses the JTAG port available on most ICs to read back memory SUVs to determine and interrogate digital fingerprints. A strong physically unclonable function (PUF) is used to statistically analyze the readback data A) to create a set of unique fingerprints for every IC, and B) to compare further interrogations to the known fingerprints for authenticating the ICs.

The Memometer is based on a generic hardware descriptive language (HDL) kernel that can read power-on SUVs of uncommitted D-FF and RAM memory values. This generic tool is simulatable, synthesizable and extensible. Since it adheres to the

IEEE 1149.x JTAG standard, it is not dependent upon any specific device or manufacturer. Figure 1.5 shows the Memometer system. In this figure, the Memometer, is accessing power-on uncommitted memory SUVs of a circuit under test (CUT). These SUVs are further used to create fingerprints and to tag ICs with these unclonable fingerprints.

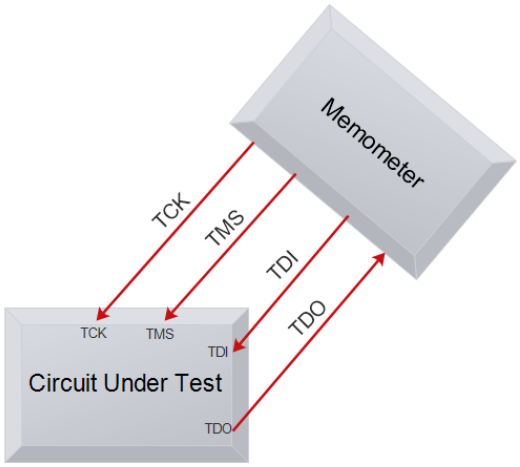


Figure 1. 5 The Memometer System

During our development, we have found that newer FPGAs come with power-on manufacturing memory preset [34]. This preset characteristic suppresses the application of memory PUFs in FPGA applications. To overcome this problem, we have created our own memory PUF using cross-coupled NAND/NOR gates mapped to FPGA look-up tables (LUTs) – as shown in Figure 1.6 – that are not confined to fixed preset SUVs. We programmed our methodology on Xilinx FPGAs. Our fingerprints are not only unique for each FPGA but also consistent (reliable) every time the device is powered on.

A weak PUF-based methodology creates and uses only a handful of fingerprints, whereas a strong PUF-based methodology uses many fingerprints per IC. Instead of having one fingerprint per device, our methodology makes provision for many (hundreds) of fingerprints per device.

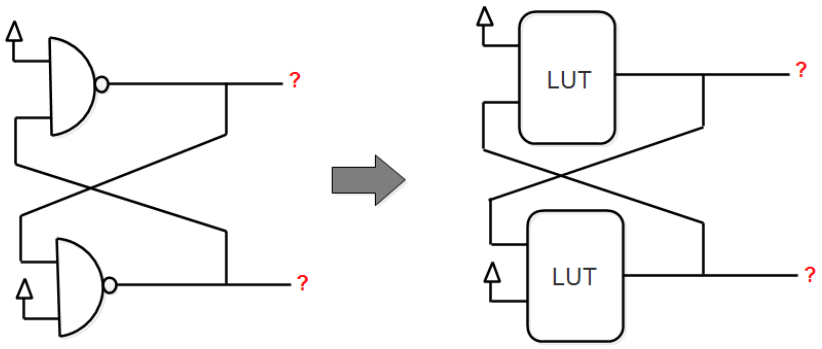


Figure 1. 6 Cross-coupled NAND gates mapped to cross-coupled LUTs

1.3 Dissertation Outline

The rest of this dissertation is organized as follows: in chapter 2, we discuss the background and related work; in chapter 3, we discuss the technical details of the Memometer and a method to overcome the power-on memory preset problem using our memory PUF methodology, allowing our methodology to be applicable to all FPGAs; in chapters 4 and 5, we discuss the results, followed by future work.

Chapter 2: Background Study

2.1 Introduction

The Memometer was developed with the intention of applicability to various ICs such as FPGAs, ASICs, and PLDs. One of the most common components in all ICs is the IEEE 1149.x JTAG standard test access port (TAP) state machine. We started our metering methodology with the intention of utilizing the JTAG TAP controller so that we could make our solution applicable to FPGAs, PLDs, ASICs, and other custom chips. The Memometer accesses the JTAG TAP controller in order to read back power-on uncommitted memory SUVs to create and authenticate unique fingerprints.

2.2 IEEE 1149.x (JTAG) Standard

The Joint Test Action Group (JTAG) developed the S*tandard Test Access Port and Boundary-Scan Architecture*, which is called IEEE 1149.x standard. This set is used to standardize the testing of digital ICs and to assemble printed circuit boards (PCBs), ensuring the integrity of those circuits and their interconnects [21]. The TAP is the key component of the test logic, and it makes use of a standardized finite state machine (FSM) to access scan registers and other components required for scan-based IC testing. Kenneth Parker, a member of the JTAG committee, stated that the "TAP controller state diagram is the fundamental road-map that all IEEE 1149.x applications must follow" [41]. Figure 2.1 shows the TAP controller state diagram. The TAP controller FSM contains 16 states. It has four required pins: test clock input (TCK), test mode select input (TMS), test data input (TDI), and test data output (TDO). These pins are used to

communicate with boundary-scan logic and other ICs that are JTAG compliant. The TMS input is used to navigate through the states on every positive edge of TCK.

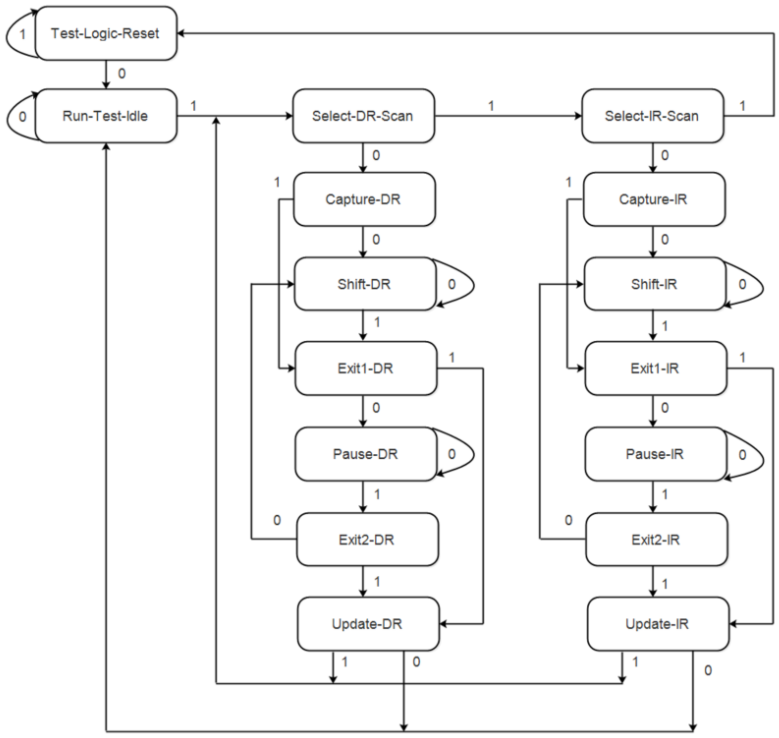


Figure 2. 1 The IEEE 1149.x JTAG standard TAP controller [41]

ICs are tested using this JTAG standard by a test logic consisting of boundary scan registers. These registers are accessible through the TAP controller. The TAP controller states are accessed by the TMS for every rising edge of the TCK. When the TMS is held high for a minimum of five clock cycles, the TAP controller enters the Test-Logic-Reset state. The Memometer accesses different instruction registers (IRs) and data registers (DRs) of the TAP controller using the TMS and TCK pins. Instructions are serially loaded into the IR through the TDI for every rising edge of the TCK, and output

data is observed for a particular DR through the TDO at the falling edge of the TCK. Figure 2.2 shows the typical JTAG compliant device architecture [41].

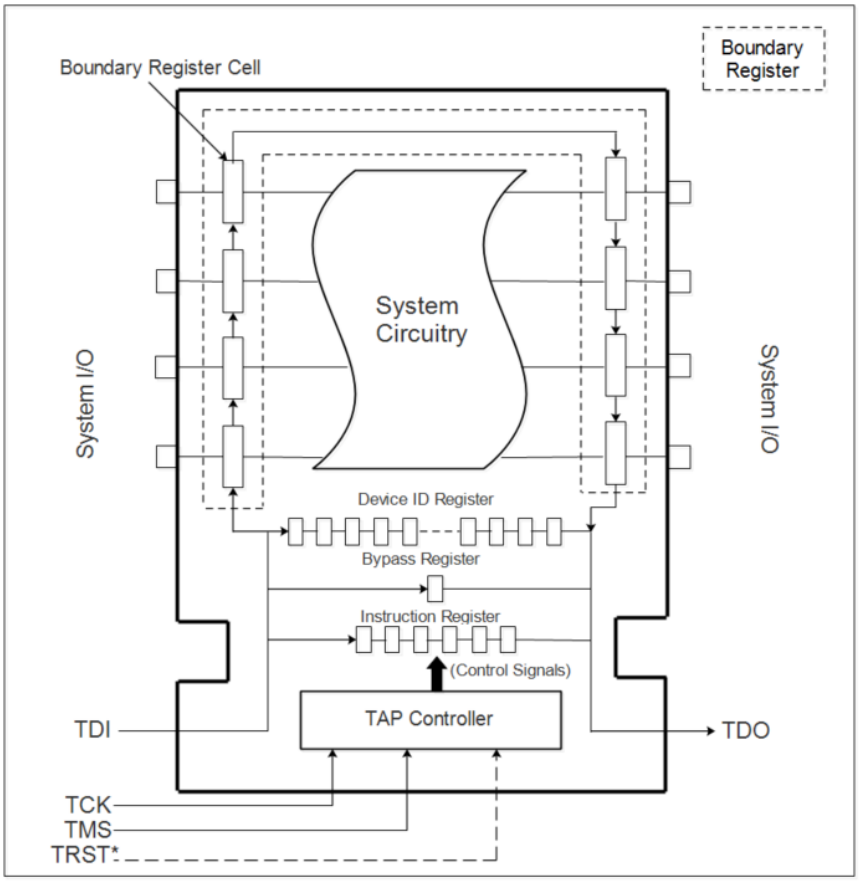


Figure 2. 2 Generic architecture of an IEEE 1149.1 (JTAG) compliant IC [41]

In addition to the DR and IR registers, other boundary-scan registers include Bypass, Device Identification, Boundary, and User Defined. Also, five other registers were added to the IEEE 1149.1 Standards in the 2013 update: ECID, Init Data, Init Status, Bypass Escape, and Reset Select [41]. Typical scan tests involve daisy chaining scan-cells and registers and controlling them with the JTAG TAP controller.

Any uncommitted (unprogrammed) scan-cells and registers' SUVs are the potential candidates for fingerprints. These SUVs are used to A) develop digital fingerprints for the ICs, and B) integrate those ICs to verify and validate the authenticity of the ICs.

2.2.1 IEEE Standard 1532

The IEEE 1532 Standard was proposed to enhance access to IEEE 1149.x. Primarily the IEEE 1532 Standard was developed to address the "In-System Configuration" (ISC) of programmable devices like FPGAs. Kenneth Parker, Editor of these standards, stated that the very first rule in the IEEE 1532 standard is "Thou shalt be compliant with 1149.x" [41]. This statement implies that the same TAP controller is used for all additional functionality that was proposed in the IEEE 1532 Standards. Also, the IEEE 1532 standard states: "The objective of 1532 standard is to enhance the user access to IEEE 1149.1 based programmable devices. Since such devices are built on the foundation of the IEEE Std 1149.1 TAP and state machine, they already comply with that standard" [20]. This methodology has become the standard for programmable ICs like FPGAs [20]. Neil Jacobson, from Xilinx, who is also the chair of IEEE 1532 standards, noted that SRAM cell-based devices are always in-system configurable and also that Xilinx FPGAs support the IEEE 1532 standard [25]. Xilinx was the first programmable device company to introduce SRAM based configurable FPGAs; afterwards, Altera (Intel) and Lattice adapted SRAM based programmability to their devices [25]. SRAM based programmable memory cells and scan test cells are key components of our Memometer system. Figure 2.3 shows the IEEE 1149.x standard compliant Xilinx 7-series FPGA architecture [60]. As shown in this figure, the TMS is controlling the states,

and different registers are accessed based on the instruction placed into the IR. The Memometer can access the JTAG TAP controller on any IC, including any programmable devices that adhere to the IEEE 1149.x standard. Some of the generic features of the 1532 standard are to configure, reconfigure, read back, verify, and erase programmable devices [24]. FPGA un-programmed power-on configuration memory SUVs can be read back using the TAP controller. These uncommitted memory SUVs can be further used to develop fingerprints.

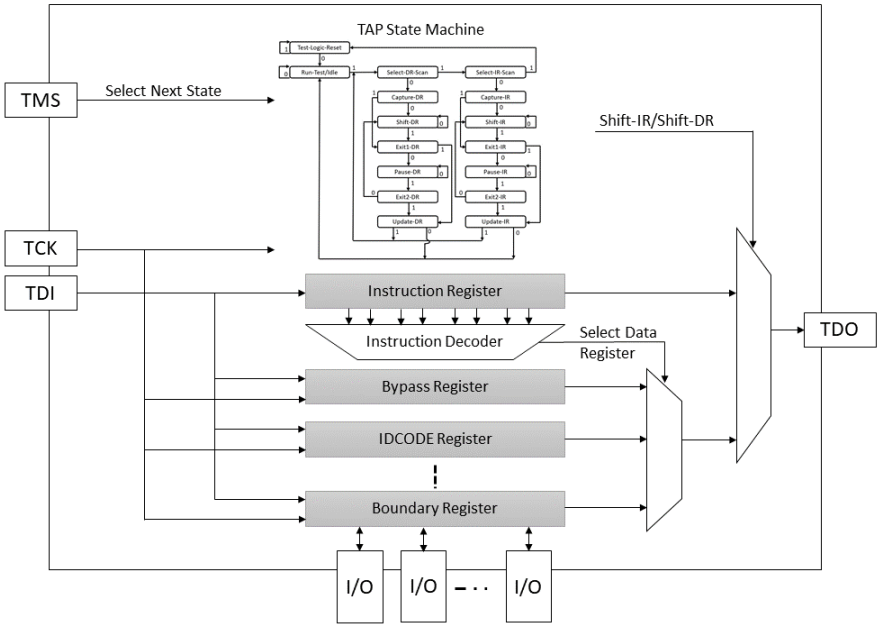


Figure 2. 3 The IEEE 1149.x Std. compliant Xilinx 7-series FPGA architecture [60]

2.2.2 JTAG Boundary Scan Chain

All ICs that adhere to the IEEE 1149.x JTAG standard can be daisy chained together as shown in Figure 2.4 [41]. In this chain all TCKs are linked together, and all TMSs are linked together. The TDI/TDO pins of the ICs are connected in a serial fashion

to form the test chain. This allows all ICs in the chain to communicate (TMS/TCK) and share data (TDI/TDO). The Memometer can be easily adapted to take advantage of this boundary scan chain structure to generate memory SUVs for the corresponding ICs. These SUVs are further used in creating fingerprints for the corresponding ICs.

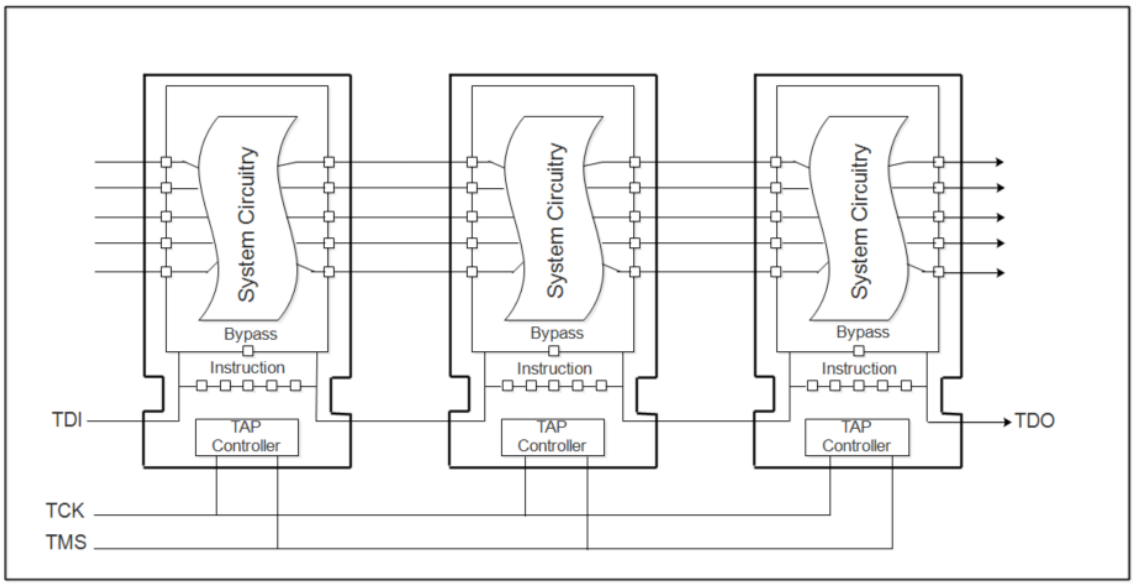


Figure 2. 4 Simple boundary scan chain [41]

2.3 Physically Unclonable Function

Identification and authentication are crucial to secure any electronic system. Embedding a unique ID can only help identify the IC, but in order to authenticate, a secret key must be embedded onto the IC itself [11]. These keys are either stored in a non-volatile memory (NVM) or battery-backed external volatile memory. Both methods are prone to be vulnerable to attackers and add additional overhead. A simple side-channel attack can reveal a lot about the IC and allow for the secret key to be stolen [29]. To overcome this issue, a new authentication method was invented that is A) extremely

hard-to-forge, B) unique to every IC ever manufactured, C) non-programmed, and D) low overhead.

A Physical Random Function, also known as a Physically Unclonable Function (PUF) is defined as a disordered physical system which, when interrogated by a challenge C_i , produces a unique response R_{C_i}) [11, 45]. The basic idea behind a PUF is that manufacturing process variations within each IC fabricated under the same mask produces unique characteristic profiles. These profiles could be turned into unclonable unique digital fingerprints. For identification and authentication, PUF challenge response pairs are initially recorded in a secure database. Later, when authenticating an IC, a random challenge is selected and applied to the IC, and the generated response is compared with the responses in the database to match [53]. Hamming distance is used to match the response. The response that has the lowest HD (usually matched to a very low threshold) is considered a match. The response that has a highest HD is considered a mismatch [19].

2.3.1 Metrics

The uniqueness and reproducibility of the PUF responses are measured by using inter-chip and intra-chip hamming distance (HD) [47, 53].

- *Inter-chip HD* is the average HD measured between the responses when the same challenge is applied to two different ICs. Ideally, it should be 50%, which means half of the bits from these two different fingerprints must be different. This measurement can also be used for analyzing two different fingerprints obtained

from two different locations within the same IC. Inter-chip HD is used to measure the uniqueness of the fingerprints.

- *Intra-chip HD* is the average HD measured between the responses when the same challenge is applied at different times. Ideally, this should be 0%, which means each fingerprint must be repeatable over time.

2.3.2 Strong PUF vs. Weak PUF

PUFs are categorized as weak and strong PUFs based on the number of challenge-response (C_i, R_{C_i}) pairs. A weak PUF has a limited number of (C_i, R_{C_i}) pairs – in most instances only one challenge per PUF [46] – whereas a strong PUF has many (C_i, R_{C_i}) pairs. Due to the high number of (C_i, R_{C_i}) pairs, often strong PUFs are best used in authentication [17]. For authentication, a strong PUF can afford to have a different challenge each time. However, since a weak PUF only has one or a limited number of (C_i, R_{C_i}) pairs, it is not advisable to use it for authentication by itself without having some type of encryption padded to it. Because of this, weak PUFs are often used as cryptographic keys in crytpo modules and/or applications [17].

Since PUFs are based on the inherent process variation of an IC, small change within its internal noise can affect the PUF response. Hence error-correction is inevitable in any PUF based implementation [46], especially when considering a weak PUF, which is mostly used in cryptographic applications. For a PUF response to be used as a cryptographic key, the generated response must be the same for each power cycle. Often this is accomplished by having some type of helper data stored in NVM. This helper data does not reveal anything regarding the PUF response. Figure 2.5 shows the PUF based

key generation [53]. In the initialization phase, syndrome/helper data is generated. This public information is then used in the regeneration phase. Any PUF circuit that will be used as a cryptographic key needs to follow this initialization / regeneration phase architecture using error-correction mechanism.

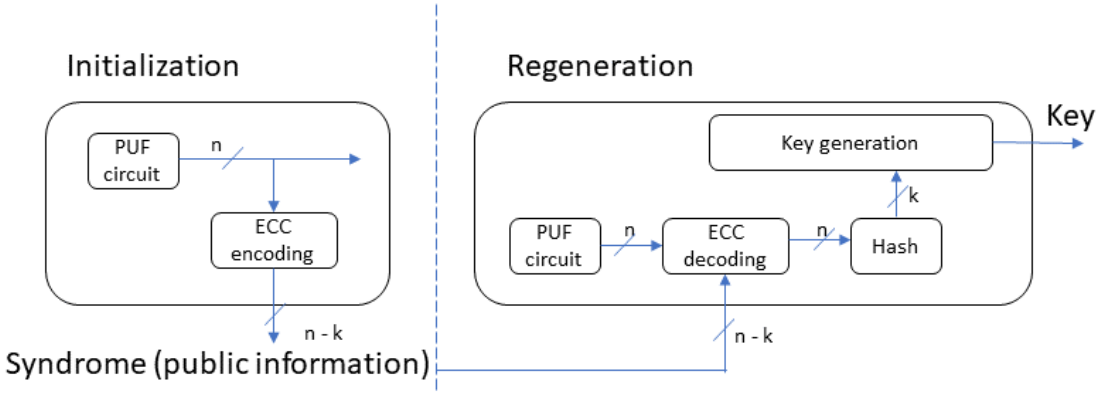


Figure 2. 5 PUF based key generation [53]

Strong PUFs, often used for authentication, can function without having an error correction. For example, to implement a low-cost authentication mechanism, a simple HD comparison is sufficient. Since a strong PUF has many (C_i, R_{C_i}) pairs, its PUF implementation can afford to have unique (C_i, R_{C_i}) pairs for every authentication.

Here is a low-cost authentication protocol mechanism involving a server and client setup using a Strong PUF [17]:

- *Step 1*: PUF-based device is manufactured.
- *Step 2*: Device's PUF (C_i, R_{C_i}) pairs are generated and stored in a secure database on the server side.

- *Step 3*: Device is placed in the supply chain.
- *Step 4*: Client picks up this device and requests server for authentication.
- *Step 5*: Server sends out a challenge.
- *Step 6*: Client generates the response and sends it to the server.
- *Step 7*: Server authenticates the response by comparing against the (C_i, R_{C_i}) pairs in its secure database.

2.3.3 Various PUF Designs

In this section we introduce a handful of PUF constructions. There are many different types of PUF designs that are available. PUF-based research has been ongoing for the past two decades [45, 17].

1) ***Arbiter PUF***: An Arbiter PUF is a type of silicon PUF based on multiplexers and an arbiter [53, 54]. Figure 2.6 shows the construction of an Arbiter PUF circuit. An Arbiter PUF circuit has multiple bit inputs X (as challenge) and one bit output Y (as response). A rising signal is given at the input of the first multiplexer. Based on challenge X, the input rising signal is raced through two delay paths. The latch outputs the response '1' when input D is faster than CLK; otherwise it is '0'. In order to generate a 128-bit output, a pseudo-random number generator (PRNG) is used as an input seed for challenges. This PRNG is given to the input X for 128-challenges to generate 128-bit response. A 64 stage Arbiter PUF design has been fabricated using TSMC 0.18 um. An inter-chip variation of 23% and intra-chip variation of 0.7% is shown by using their implementation. An ideal symmetric layout construction would have given a more ideal 50% inter-chip variation.

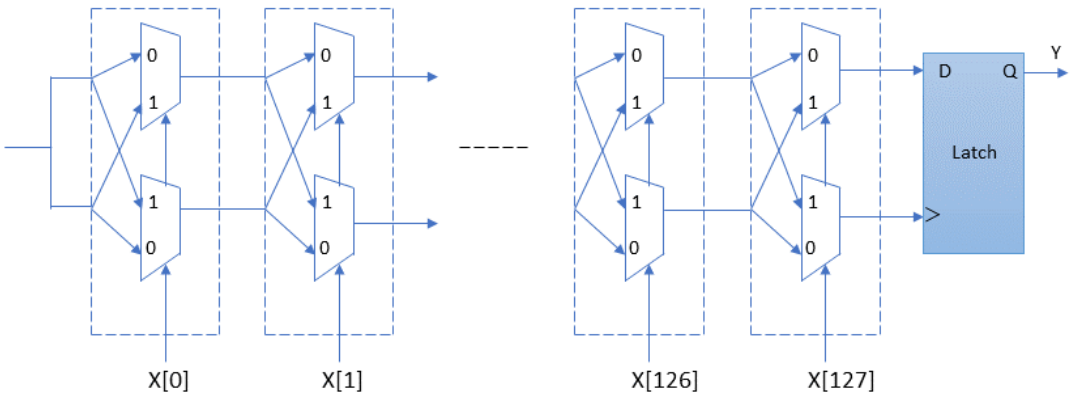


Figure 2. 6 Arbiter PUF circuit [53]

2) ***Ring Oscillator PUF***: As noticed in the Arbiter PUF design, in order to get higher inter-chip variation, a more symmetric layout construction was needed. Hence the Ring Oscillator (RO) PUF was invented as a complimentary design to the Arbiter PUF. Figure 2.7 shows the ring oscillator (RO) PUF construction [53]. The RO PUF is a delay based PUF that does not require a rigid layout and is easier to implement. The basic idea behind this construction is that each RO oscillates with a unique frequency, due to manufacturing variations, from wafer to wafer and also from different positions within the same wafer. A counter is used to compare those frequencies to generate the output bit. Multiple bits are generated based on different RO frequency comparisons. 1024 ring oscillators are laid in a 16-by-64 array on 15 Xilinx Virtex4 LX25 FPGAs. The RO design consists of five inverters and one AND gate, implemented using FPGA look-up tables (LUTs). In order to overcome the environmental variations, ROs that are far apart were

chosen for reliable bit generation. For a 128-bit PUF response, the inter-chip variation is 46.15% and an intra-chip variation is 0.48%.

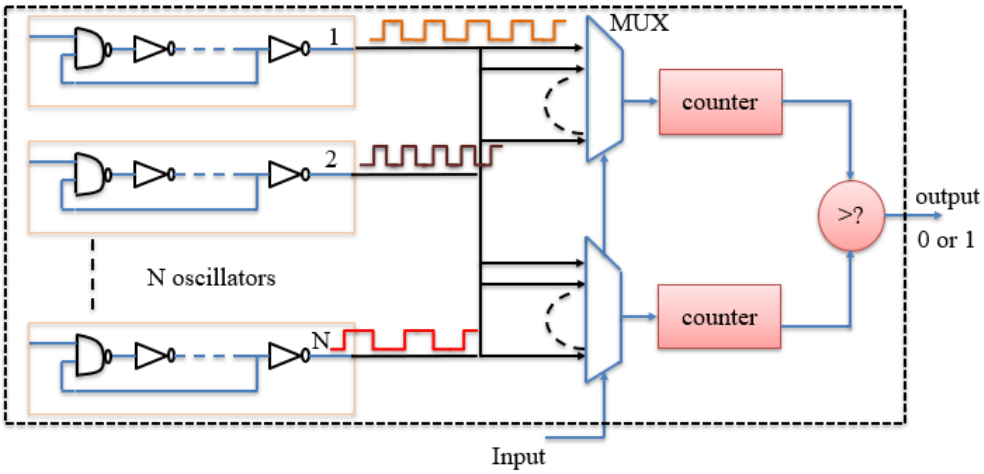
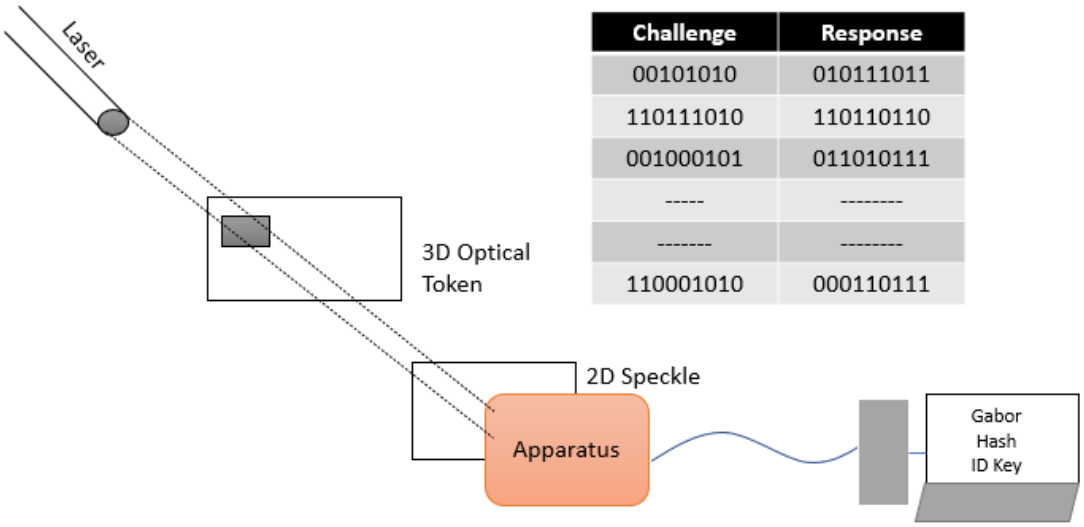


Figure 2. 7 Ring Oscillator PUF circuit [53]

3) ***Optical PUF***: The first recognized optical PUF was published as a physical-one-way-function (POWF) [40, 45]. In cryptography, algorithmic one-way hash functions are supposedly irreversible, harder to break, and require exponential time. Due to the increase in technological computer infrastructure, parallel processing, and quantum computers, there has been no proof that these types of attacks do not exist [40]. Instead of using number theory to implement one-way hash functions, the authors of "Physical One-Way Functions" [40] used a 3D in-homogeneous structure token. These token structures were made by applying optical-grade epoxy to the micron-scale glass spheres. As shown in Figure 2.8, a laser is used to generate a 2D speckled image using a 3D in-homogeneous token. This 2D image is turned into a 1D key using a multi scale Gabor Transform. The laser XY location and polarization is used as an input challenge and the 2D

speckle is the response generated which is turned into a 1D key [17]. In other words, different incident angles of the laser are used to generate different keys. A 2400-bit key has a mean inter-variation distance of 0.5, which is a 1200-bit difference, and an intra-variation distance of 0.25 equaling 1800 bits being matched correctly.



Challenge	Response
00101010	010111011
110111010	110110110
001000101	011010111
-----	--------
-------	--------
110001010	000110111

Figure 2. 8 Optical PUF setup [40, 45]

4) ***SRAM PUF***: The Arbiter PUF and RO PUF were explored based on the path delay variations, hence they are delay based PUFs. On the other hand, the SRAM PUF was explored based on MOSFET threshold voltage differences [52]. K. Loftstrom et. al. [37] explored the IC identification (ICID) using the MOSFET device threshold voltages as shown in Figure 2.9. In ICID, due to the inherent device mismatch, a unique sequence of

random voltages across the load are converted into a unique identification. An auto-zeroing comparator is used to convert the random voltages to binary sequence. Due to different random dopant atoms that exist in different locations, different ID bits are generated from IC to IC and also within the same IC.

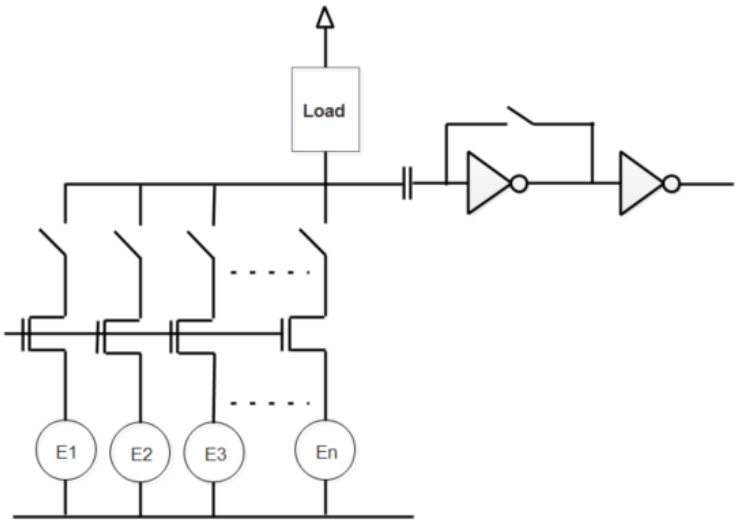


Figure 2. 9 ICID circuit [37]

Based on the idea of difference in threshold voltage, Y. Su et al. [52] constructed a positive feedback loop circuit using cross-coupled NOR (NAND) gates that function in the same way as an SRAM cell. Figure 2.10 shows the six transistor SRAM cell and Figure 2.11 shows the memory PUF based on the cross-coupled NOR design. This cross-coupled circuit is used to create unclonable IDs. In this circuit, instead of using an amplifier and comparator, they used a positive feedback circuit for digitization. When the reset is pulled low, the state of this cross-coupled circuit is determined by the intrinsic mismatch. Around the same time, a few researchers were exploring different circuits that

are already available intrinsic to the IC to be used as a PUF, which has led to exploring SRAM as that potential candidate [13, 19].

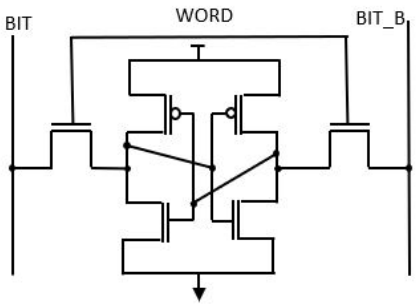


Figure 2. 10 SRAM cell transistor diagram [25,58]

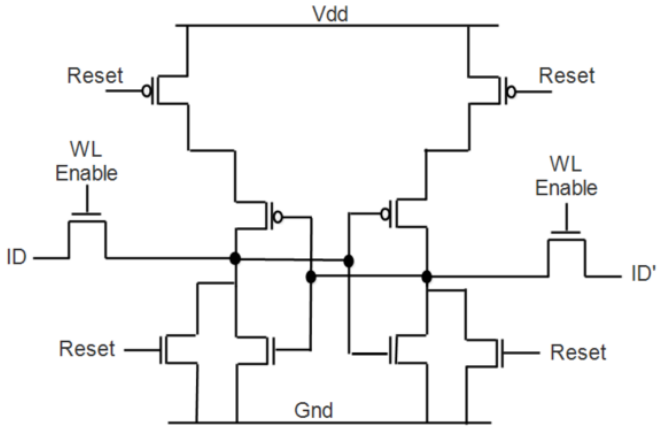


Figure 2. 11 Positive feedback ID circuit [52]

SRAM is one of the most widely available elements in any digital circuit. It has been noted that, because of its cross-coupled design nature, each SRAM cell is powered-on to a random start-up value [4, 13, 19]. SRAM cells are usually designed with minimum size and are balanced or symmetric relative to drive strength. At power-on, metastability makes it difficult to detect what state the cells will take: logic '1' or logic

‘0’. Due to doping variations and parasitic differences within the same IC device, some cell SUVs will be powered on to their preferred state – either logic ‘1’ or logic ‘0’ – while others will power on unpredictably to either logic ‘1’ sometimes and logic ‘0’ other times. This metastability is caused by different parameters as shown in Figure 2.12. Based on the sensitivity, the SRAM cell is classified as a non-skewed, partially-skewed, or fully-skewed cell [4, 19].

- ***Non-skewed cell***: This type of cell does not have major mismatch between its cross-coupled inverters. The internal noise present at the power-up will determine its state as either ‘0’ or ‘1’ unpredictably. These types of cells are also called neutrally-skewed cells [19].
- ***Fully-skewed cell***: This type of cell has higher mismatch between its cross-coupled inverters which eventually leads to always taking up its preferred state – either logic ‘1’ or ‘0’. If it is a 0-skewed cell, it takes a preferred state as logic ‘0’ at power-up. If it is a 1-skewed cell, it takes a preferred state as logic ‘1’ [19].
- ***Partially-skewed cell***: This type of cell has a preferred state, but unlike a fully-skewed cell, any smaller external variation can cause this cell value to flip.

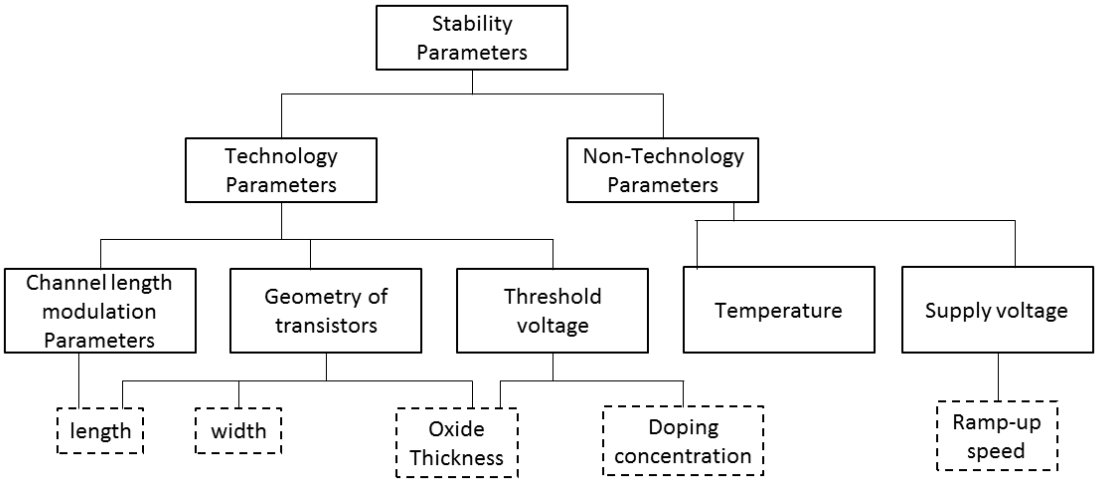


Figure 2. 12 SRAM PUF sensitivity parameters [4]

An SRAM fingerprint is a collection of SRAM start-up values used for the purpose of authentication. An SRAM fingerprint is further categorized as a latent or known fingerprint [19].

- ***Latent Fingerprint***: A latent fingerprint is defined as SRAM start-up values at any given time. In other words, when an IC is powered-on, the SRAM SUVs that are acquired from that IC are categorized as a latent fingerprint. This fingerprint consists of a collection of non-skewed cells, fully-skewed cells, and partial skewed cells.
- ***Known Fingerprint***: A known fingerprint is estimated based on the most likely power-up state of SRAM cells. In other words, a collection of latent fingerprints is taken, typically an odd number, and the most likely power-up state for each cell is determined by taking an average of the odd number of trials. Logic '1' is assigned to a particular cell if its probability of power-up value is greater than 0.5;

otherwise, it is logic ‘0’. Hamming distance comparison is used to identify one IC fingerprint from another [19]. However, several different statistical methods can be used to create different identification methodologies.

2.3.4 PUFs in the Commercial Market

PUFs are gaining popularity as a go-to solution for hardware root-of-trust (RoT). This section helps us understand the validity of PUFs in the commercial market. Here are a few PUF based commercial companies that are in the market currently (Note: This is not an exhaustive list).

I. ***SiidTech’s ICID***: ICID stands for Integrated Circuit IDentification [37]. ICID creates an unclonable unique identifier based on mismatch MOSFETs as shown in the Figure 2.13. An ICID circuit produces logic ‘1’ for a positive offset and ‘0’ for a negative offset. SiidTech is an Oregon based company that provides ICID licensing [51]. They provide this ICID circuit as a full custom layout standard cell which can be integrated into any custom CMOS digital IC design. The ICID circuit is used for identification, generation of cryptographic keys and one-time pads [51].

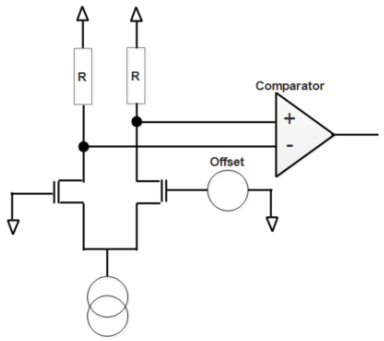


Figure 2. 13 SiidTech’s ICID PUF [51]

II. ***Intrinsic-ID's SRAM PUF*:** Intrinsic-ID is a Netherlands (HQ) based company that has patented SRAM PUF technology [22]. It is a spinoff of the Royal Philips Research group. As discussed in the previous section about SRAM PUFs, Intrinsic-ID uses the SRAM start-up values to create fingerprints to secure ICs. Figure 2.14 shows Intrinsic-ID's SRAM PUF methodology [23]. When the device is powered on, due to the intrinsic unique process variation present within each memory element, these SRAM cells are powered-on to their unique preferred state, either logic '1' or logic '0'. To generate a reliable key from the SRAM SUVs, a fuzzy extractor is used. Thus, the generated unique key is used in cryptography applications like AES. It has been shown that each chip generates a unique key.

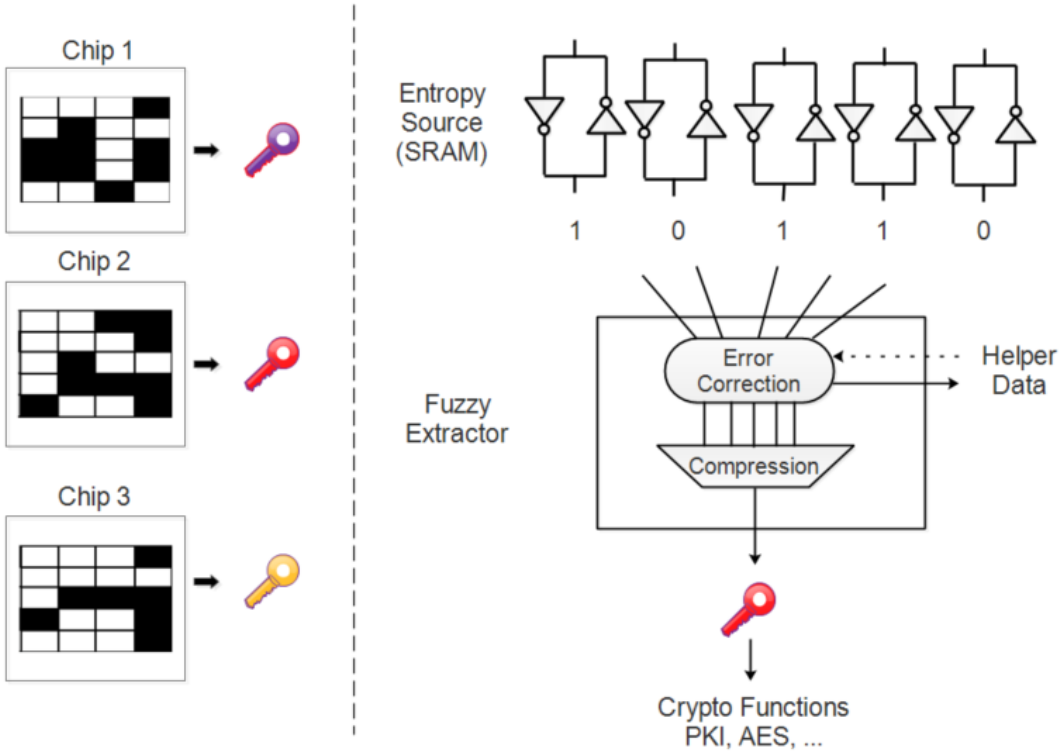


Figure 2. 14 Intrinsic-ID's SRAM PUF Methodology [23]

III. ***PUF Security's Neo PUF:*** PUF security is a Taiwan based company [49], which is a subsidiary of eMemory [6]. Their Neo PUF is based on capitalizing on randomness present in the MOSFET oxide properties [59]. Figure 2.15 shows the Neo PUF's circuit diagram. When a high voltage of 5.5V is applied to the AF0/AF1 simultaneously, due to the inherent thickness of the oxide layer, it starts breaking down on either side of the AF0 or AF1. The PUF output is obtained by converting this oxide breakdown into binary values. An output of logic '1' or '0' is determined by using a voltage comparator. A Vref (reference voltage) is used for binarizing. When the output BL voltage is higher than Vref, then it is noted as

logic ‘1’; when the voltage is lower than Vref, it is logic ‘0’ [59]. They have shown that their PUF design does not need an error-correction and gives an ideal hamming distance.

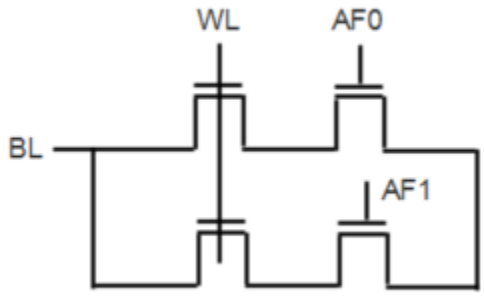


Figure 2. 15 PUF Security (eMemory)’s Neo PUF circuit [59]

2.4 Hardware Metering

A state-of-the-art fabrication facility is required to build modern complex ICs. Building such a fabrication facility is an expensive process. Because of this, many semiconductor companies have shifted toward a horizontal business model also known as a “fabless” model [32, 31]. In a “fabless” model, an asymmetric relationship exists between the design house and the manufacturing facility, where design and fabrication are done at different facilities. Since the design house provides all the necessary files to fabricate an IC, it is extremely low overhead for the fabrication facility to overproduce ICs.

2.4.1 Passive Metering

Hardware metering helps in achieving the post-fabrication control of ICs for the design house by providing a unique tag for every IC that is manufactured under the same mask [31, 32]. In other words, hardware metering is comprised of a set of tools, methods, and protocols that enable the design house or IP owner to track ICs post-fabrication [47]. Metering is further classified into passive and active. In passive metering, a chip is tagged with a unique unclonable identifier. This identifier is used in identification of ICs [31]. Whereas in active metering, in addition to tracking passively, it can also help with enabling/disabling IC functionality and controlling/preventing the ICs from entering the supply chain [31]. Passive metering is further classified into extrinsic and intrinsic [31]. In extrinsic passive metering, unique tags are produced based on additional logic or modification to the design during the IC synthesis step. Intrinsic passive metering does not rely on additional hardware to create unique IDs. Hence intrinsic passive metering can be used for tagging legacy ICs [31]. Passive metering paired with an intrinsic PUF yields a robust, low-cost, low-overhead supply-chain fingerprinting methodology.

2.4.2 Active Metering

Active metering is also used as a type of logic locking mechanism. In active metering, when an IC is fabricated, initial functionality of the IC is locked. The manufacturer needs to contact the IP owner to unlock each IC [48]. The IC generates a unique unclonable fingerprint upon fabrication. The manufacturer generates this initial raw fingerprint and sends it to the IP rights owner. Then the IP rights owner sends back an activation code to activate the chip based on the fingerprint. Then the activated chip is

placed in the supply chain. This methodology makes provision for the IP rights owner to have control over exactly how many chips exist in the supply-chain.

Another active hardware metering approach uses finite state machines (FSMs) [32, 33]. The basic idea is that modern designs consist of a large number of FSMs with unused state combinations. These unused states can be laser burned in the post-processing step in such a way that each design has a slightly different control path. Computer simulations can be used to generate laser burn patterns that are unique for each design. Figure 2.16 shows the FSM based metering system [31]. The original FSM is modified to add (or replicate) a few other states. A PUF based key is used as a transition from the redundant state to the original state. This type of FSM based locking/unlocking mechanism is not only used to secure ICs but also used to secure IP cores [31].

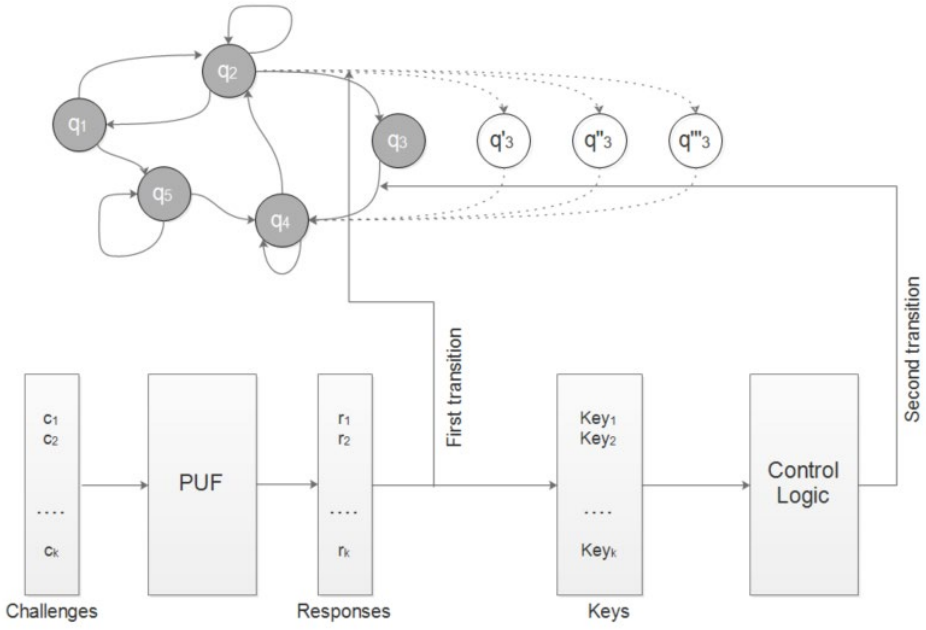


Figure 2. 16 Active Metering based on FSM [31]

2.5 Memory-Based Metering

The Memometer uses hardware metering methodology to maintain control and accountability of ICs. Figure 2.17 shows the PUF-based authentication for securing ICs in the supply chain [53]. PUF challenge-response pairs (C_i, R_{C_i}) are stored in a secure database at the design house or with the IP rights owner. When an IC is picked up from the supply chain, the (C_i, R_{C_i}) pairs are compared with those in the secure database for authentication. Instead of having one (C_i, R_{C_i}) pair device, the Memometer uses many (C_i, R_{C_i}) pairs per device for identifying untrusted ICs, making it a strong PUF-based methodology.

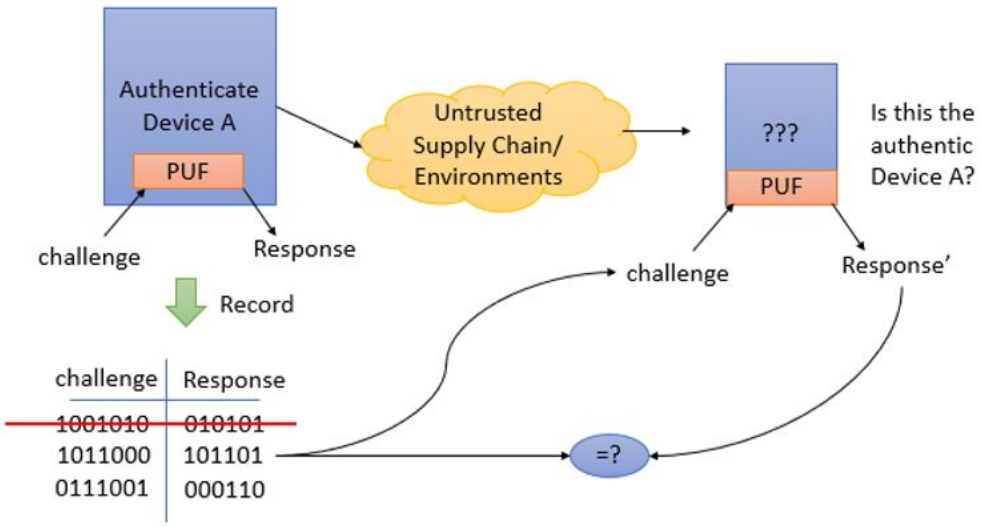


Figure 2. 17 PUF-based authentication [53]

CHAPTER 3: Technical Details

3.1 Goals

The overall goal for the Memometer project is to develop a tool that is capable of generating digital signatures using programming memory by accessing the JTAG interface on ICs. These generated digital fingerprints are further used for interrogating and verifying the authenticity of ICs within the supply chain. This goal was executed in two stages:

- *Stage 1*: A configurable generic firmware interface was developed to access the JTAG port available on ICs. We used an HDL to develop this kernel. The Memometer kernel is completely platform independent. It is simulatable and synthesizable on any commercially available CAD tool. It is compatible with any FPGA and ASIC vendor that adheres to the IEEE 1149.x (JTAG) standard. This kernel is used to read the un-programmed memory SUVs for fingerprinting.
- *Stage 2*: A strong PUF statistical model was developed for (C_i, R_{C_i}) pairs. This statistical model was applied to the memory SUVs to create digital signatures. These signatures were further used for interrogation. Instead of creating one fingerprint per device, the Memometer creates hundreds of fingerprints per device.

3.2 Memory PUF-Based Fingerprinting

Memory is a core component of all digital electronic systems. Memory types include volatile and non-volatile. Anti-fuse cells and flash electrically erasable and

programmable memory (EEPROM) cells are two of the widely used non-volatile memory cells. Similarly, static random-access memory (SRAM) cells are the most used volatile memory cells. Traditional cryptographic systems depend on non-volatile memories to store secret keys. Storing a key permanently on the device is no longer a reliable approach due to the increase in side-channel security attacks. Therefore, researchers have started studying key generation techniques using other sources and other ways to secure/authenticate memory chips to avoid counterfeit parts [4, 19, 28].

The memory based PUF is one of the emerging PUF methodologies in the field of hardware security. Creating digital signatures using memory elements is becoming an opportune solution in the field of security [14]. SRAMs and D-FFs are two widely used memory elements in digital systems. Memory PUF research shows that when a cross-coupled memory structure is manufactured for minimum size, as shown in Figure 3.1, the relative drive strength and doping levels are usually balanced. When these devices are powered on, before programming any value, the metastability property of balanced memory elements leads to random start-up values (SUVs). Instability within these cross-coupled components is due to several technological and non-technological parameters, such as probabilistic geometry of transistors, inexact threshold voltages, and channel length modulation [4]. Because of these varying parametric values, some memory cells always power on to a specific state, whether logic ‘1’ or logic ‘0’; that is, 100% of the time these cells are powered on to the same logic value. However, other cells fluctuate between logic ‘1’ or logic ‘0’ for each power cycle. We use the stable memory SUVs for our fingerprints.

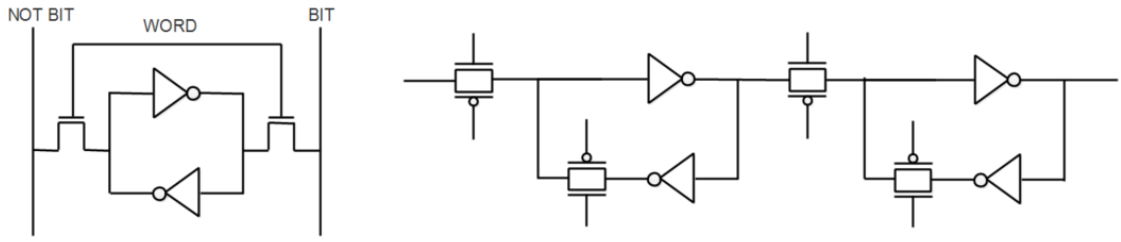


Figure 3. 1 Cross-coupled based memory elements: SRAM and D-FF [25, 58]

An IEEE 1149.x standard compliant Xilinx FPGA device architecture is shown in Figure 2.3. In this architecture, a state machine is accessed through the four JTAG pins. Different instructions can be placed in the instruction register by navigating into the Shift IR state. Different instructions enable us to access different functionalities of the FPGA. Key components of standard SRAM cells as well as most JTAG compliant scan cells are cross coupled inverters. Figure 3.2 shows a typical scan-cell design. Before programming it to a certain value, these cross-coupled memory cells are powered-up to random SUVs. These SUVs can be accessed via the JTAG port available on most ICs. Once these SUVs are read from the device they are stored on a secure database. A memory PUF statistical model is applied on these SUVs to create fingerprints.

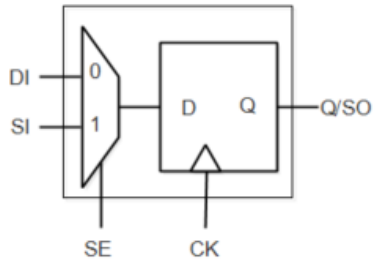


Figure 3. 2 A typical scan-cell design

The Memometer prototype system was tested using Xilinx FPGAs. The Memometer on the control FPGA accesses the circuit under test (CUT) FPGA to read back the power on memory SUVs. The Memometer kernel is designed using a hardware descriptive language (HDL) mapped on to a control system. The control FPGA uses 4 wires (TMS, TCK, TDI, and TDO) and a common ground to access the JTAG port available on any CUT IC to read back the balanced, cross-coupled memory SUVs. As we mentioned in our background study, the SRAM based configuration is most used for programmable devices. When a programmable memory device is powered on, before loading any configuration file, the Memometer accesses the TAP controller on its device to read back their unconfigured memory SUVs. The Memometer HDL code sends a series of TMS, TCK, and TDI bitstreams to access the IRs and DRs through the TAP controller. The instruction OPCODE is placed into the IR after navigating to the Shift_IR state. Then the TAP controller is brought to the Shift_DR state to access the memory SUVs. These values are captured through the TDO at the falling edge of the TCK. Once these memory values are stored on a computer, a strong PUF statistical model is applied on these SUVs to create the fingerprints.

During our analysis, we found that newer FPGAs come with power on memory preset by the manufacturer. It has been mentioned that “SRAM memories in most FPGAs are forcibly set to a known state upon startup” [34]. In other words, most newer FPGAs come with manufacturing memory preset of the SRAM and D-FF cells. This issue makes it impossible to use memory cell SUVs as fingerprints on contemporary FPGAs. We have overcome this challenge by developing our own memory PUF that does not preset to a particular value, making our Memometer applicable for all FPGAs irrespective of the technology or manufacturer.

3.3 Memory PUF for Contemporary FPGAs

As we discussed in the previous section, our original approach was targeted for any device that is JTAG IEEE 1149.x compliant, but during our development we found that many newer FPGA families come with power-on memory preset. To overcome this application problem, we have developed our own memory PUF methodology by using cross-coupled LUTs for memory cells that are logically equivalent to the cross-coupled structures found in most SRAM and D-FF cells. Our LUT based memory cell is designed by mapping cross-coupled NAND gates to cross-coupled FPGA LUTs, as shown in Figure 1.6. One of the challenges of this design is balancing feedback path delays since the FPGA routing paths are proprietary and not easily accessible to a designer. If the feedback routing paths are not matched, then the memory element produces known values instead of random values. After many experiments, we found a way to balance the feedback delay paths, resulting in unique random SUVs for each FPGA. Our preliminary

testing has shown that our fingerprints are unique from one IC to another and are repeatable every time the IC is powered on.

Memory PUF research has shown that 64 bits are enough to differentiate between all existing ICs (264 unique signatures) [19]. Therefore, to test the proof of concept, we programmed 64 of these balanced cross-coupled LUT based memory elements on ten Xilinx Artix-7 FPGAs. The same programming (*.bit) file was used to program all ten FPGAs. Each FPGA gave us a unique 64-bit memory signature.

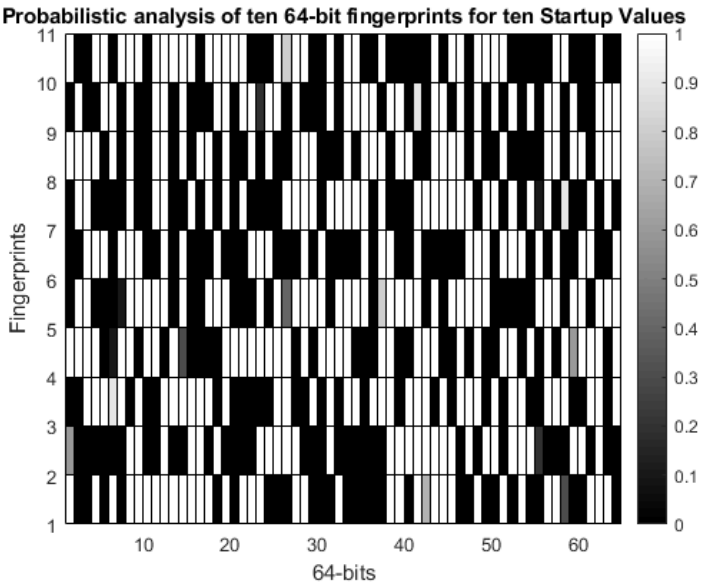


Figure 3. 3 The probability analysis of a 64-bit memory signature powering up to 1 on ten FPGAs.

Figure 3.3 shows the probability analysis of this 64-bit fingerprint powering-up to logic '1.' Each row in this figure corresponds to a unique 64 bit fingerprint from a different FPGA. This experiment was performed on ten FPGAs for ten power cycles with 120 seconds between each power cycle.

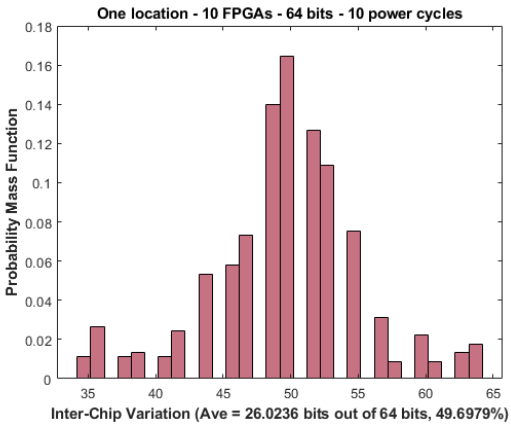


Figure 3. 4 Probability distribution of inter-chip variations of a 64-bit memory signature on ten FPGAs

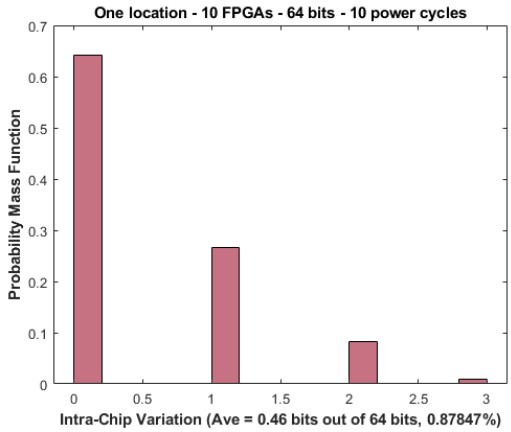


Figure 3. 5 Probability distribution of intra-chip variations of a 64-bit memory signature on ten FPGAs

Figures 3.4 and 3.5 show the probability distribution of the inter-chip and intra-chip HD of this 64-bit fingerprint. In Figure 3.4, the x-axis represents the number of PUF output bits that are different from one FPGA to another, and the y-axis represents the probability. This analysis is performed by testing 550 pair-wise comparisons. For this set of data, an inter-chip HD of 49.69% indicates that an average of 26 bits out of 64 bits are

different from one FPGA to another FPGA. Similarly, Figure 3.5 shows the intra-chip HD of 0.87%, indicating that an average of 0.46 bits out of 64 bits are changing over time.

As we mentioned earlier, we wanted to create many fingerprints per IC. To test this hypothesis, we started looking into creating fingerprints from different locations within the same IC. For our preliminary testing we programmed 64 of our memory PUF cells in 10 different locations within the same FPGA to analyze ten 64-bit fingerprints. This experiment was performed for ten power cycles with 120 seconds between each power cycle. Figures 3.6 and 3.7 show the probability distribution of the inter-chip and intra-chip HD of these 64-bit fingerprints on one FPGA taken from ten different locations. Even within a single FPGA, the data shows that we were able to achieve an average inter-chip HD of 49.74% with an average of 26 bits out of 64 bits that are different from one fingerprint to another. Likewise, intra-chip HD shows that an average of 0.73% with an average of 0.38 bits out of 64 bits are changing over time.

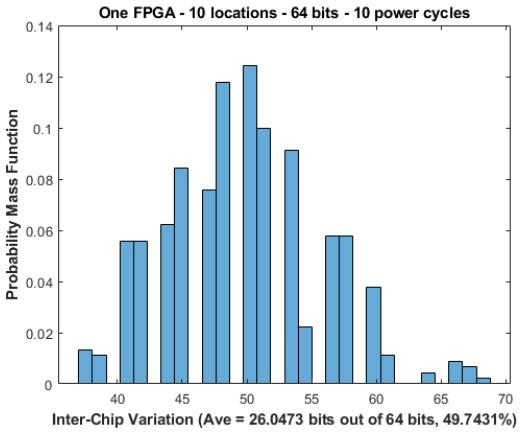


Figure 3. 6 Probability distribution of inter-chip variations of ten 64-bit memory signatures on one FPGA

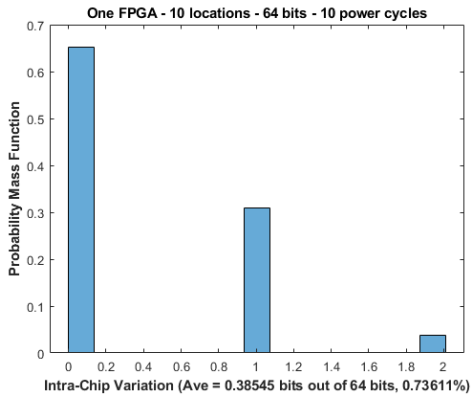


Figure 3. 7 Probability distribution of intra-chip variations of ten 64-bit memory signatures on one FPGA

Even though we have used a NAND gate-based memory PUF design, NOR gates are just as feasible as shown in Figure 3.8. When using NOR gates, the free end of the NOR gate input must be connected to logic "0" instead of logic "1." When both free ends of the cross-coupled NOR gates are connected to logic "0," the output value becomes unpredictable. At power-on, these SUVs could be used to create fingerprints just as we used NAND gates to create fingerprints.

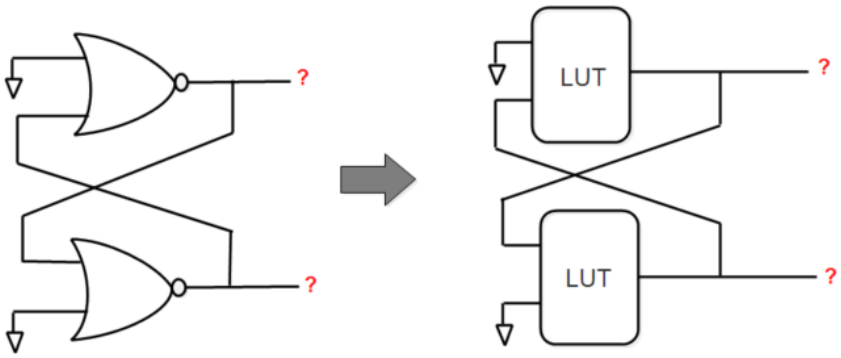


Figure 3. 8 Cross-coupled NOR gates mapped to cross-coupled LUTs

To create hundreds of fingerprints within the same IC, our approach was analogous to a reflective PUF or optical PUF. For example, reflective PUFs are used in identifying missiles [45]. A light scattering particle is sprayed on to the missile. An inspector records the images of this particle by illuminating it at different angles as shown in Figure 3.9. Each angle of incidence gives a unique pattern. When authenticating, the interference pattern is measured and compared to the one recorded image in the database. A similar idea is presented in the optical PUF as well. As we discussed in the background study, Pappu et al. proposed the first optical PUF as a strong PUF [40, 45]. The basic idea of the optical or reflective PUF methodology is that when cataloging and interrogating different angles of incidence, we end up with an endless supply of (C_i, R_{C_i}) pairs. These hundreds of (C_i, R_{C_i}) pairs are used for authentication.

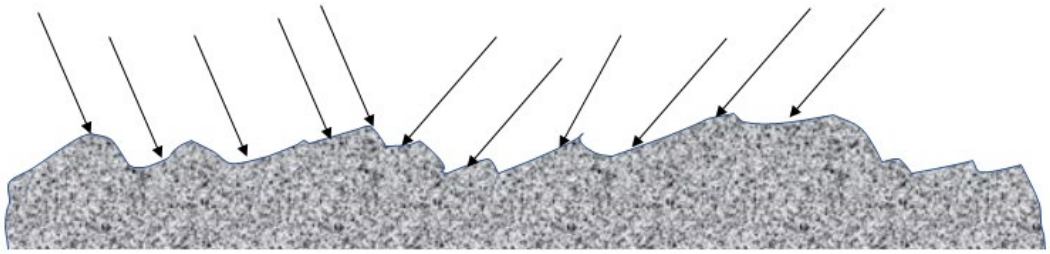


Figure 3. 9 Reflective PUF: different angle of incidence provides unique patterns [45]

The Memometer uses a similar principle by combining different combinations of memory cells from different locations, providing hundreds of possible unique fingerprints. More details of this implementation are mentioned in the results section.

3.4 Artificial Aging

Aging (or ageing) analysis of the memory PUF is performed to predict the reliability of the memory SUVs since these memory SUVs are used in developing unique fingerprints. In this research, we are investigating how these SUVs change as the device ages. The most dominant aging effect in silicon aging is the negative bias temperature instability effect (NBTI) [16]. NBTI induces bias on the PMOS transistor's absolute threshold value; in this case, this will lead to bit errors in the SUVs. NBTI occurs when the silicon is subjected to higher voltage and higher temperature stress. The acceleration factor (AF), as shown in equation 3.1, is used to calculate the amount of acceleration to which the ICs are subjected [38]. α is the gate voltage exponent, n is the time exponent, E_{aa} is the activation energy, k is the Boltzmann's constant, $V_{nominal}$ is the nominal voltage, $T_{nominal}$ is the nominal temperature, V_{stress} is the higher stress voltage, and T_{stress} is the higher stress temperature. More details of the test setup and the parameter values are described in the results section.

$$AF = \left(\frac{V_{stress}}{V_{nominal}}\right)^{\frac{\alpha}{n}} . \exp\left(\left(\frac{E_{aa}}{k}\right) . \left(\frac{1}{T_{stress}} - \frac{1}{T_{nominal}}\right) . \frac{1}{n} \right) \qquad (3.1)$$

CHAPTER 4: Results

4.1 Memometer for IEEE 1149.x (JTAG) Devices

As mentioned earlier, we wanted to create a tool that could access the JTAG port on any IC regardless of the technology or manufacturer. The Memometer tool was implemented using a generic HDL. In our case, we have used VHDL; nonetheless, Verilog is just as feasible. Our generic tool is synthesizable and simulatable on any commercially available CAD tools.

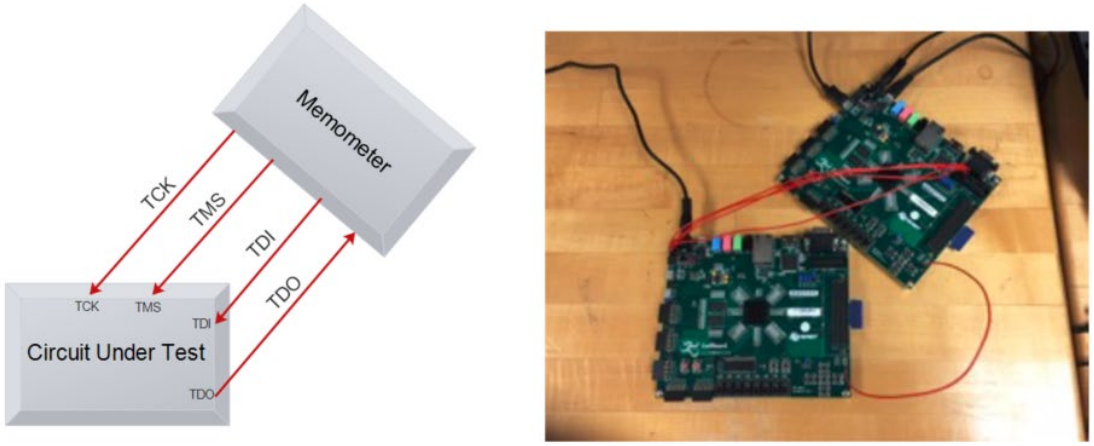


Figure 4. 1 The Memometer prototype on Xilinx FPGAs

Figure 4.1 shows the Memometer prototype on Xilinx Zedboards. The Memometer tool is mapped to the control FPGA to access the CUT. In our case, we used an FPGA as the CUT, but any IC that adheres to the IEEE 1149.x (JTAG) standard could be used. The Memometer accesses the TAP controller on any IC using readily available JTAG pins (TCK, TMS, TDI, and TDO). The control FPGA sends a series of commands

to access the TAP controller on the CUT using these four pins. For every rising edge of the TCK, corresponding TMS bitstream values are sent.

Initially, the TAP controller is brought to the Test-Logic-Reset state by raising the TMS for at least five TCK cycles. Then, the TAP controller is brought to the Shift_IR state in order to place instruction bits which access the corresponding DRs. Next, the TAP controller is brought to the Shift_DR state to shift data out. Shifted output data is observed at the TDO pin at the falling edge of the TCK. Statistical analysis is performed on this extracted data to create unique fingerprints. SRAM is the most common traditional programmable memory for FPGAs [25]. When powered on, the Memometer can read back uncommitted programmable memory using the JTAG port. A strong memory PUF statistical model is applied to this data to create unique fingerprints. Uninitialized built-in self-test (BIST) scan cells are also good candidates for fingerprints.

Our Memometer HDL kernel is adaptable, extensible to any existing digital system. Our code is low overhead, using only 62 / 53,200 LUTs on a Xilinx Artix-7 FPGA synthesized by Xilinx Vivado. Our prototype system can access the JTAG port with a clock speed of up to 10 MHz. No additional equipment is necessary other than the four wires to connect the JTAG port.

4.2 Memometer for Memory Preset FPGAs

As mentioned earlier, because of the issue with memory preset for newer FPGAs, we have created our own memory cells using LUTs that do not confine to fixed SUVs. We mapped 5,180 cross-coupled LUT based memory elements on ten Xilinx Artix-7

FPGAs. The same programming (*.bit) file was used to program all ten Xilinx FPGAs. On average, 97.08% of the bits are stable, that is, 5028.8 out of 5,180 bits are stable. Figure 4.2 shows the probabilistic analysis of these memory SUVs powering up to logic '1' for ten power cycles on an FPGA. For this particular FPGA, out of 5,180 SUVs, there are 5032 stable values. These stable values are consistently logic '1' and logic '0' for each power cycle. Figure 4.3 shows these stable and unstable values. All the stable 5032 values are shown in the yellow and the 148 unstable values are shown in blue.

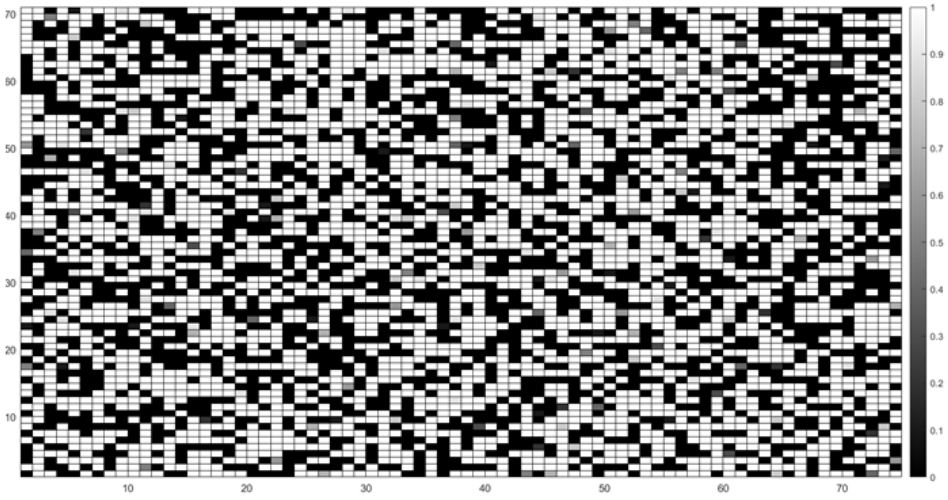


Figure 4. 2 The probabilistic analysis of over 5180 LUT based memory PUF SUVs powering up to 1 on a Xilinx FPGA

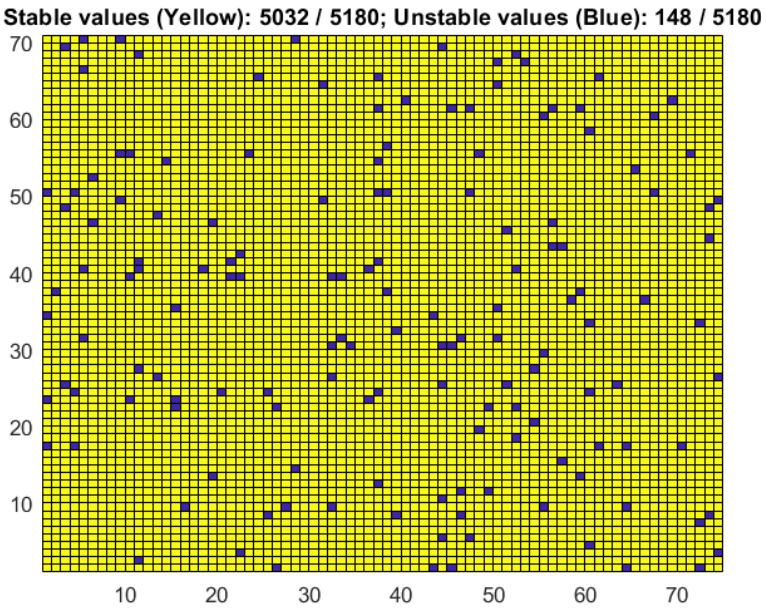


Figure 4. 3 Stable and unstable values on a single FPGA

Table 4.1 shows the summary of inter-chip and intra-chip HD for different locations on each FPGA. Each line in Figure 4.1 is 74-bits. In one FPGA we analyzed a total of 70 of these 74-bit fingerprints. Inter-chip and intra-chip HDs are measured on these seventy fingerprints per FPGA to determine the uniqueness and reliability of these fingerprints. Table 4.2 shows the overall analysis of stable values on each FPGA. For each FPGA, an average inter-chip and intra-chip HD of these seventy fingerprints are very close to the ideal 50% and 0% respectively. The average inter-chip HD is 49.12% with an upper and lower bound of 49.53% and 48.80%. For a 74-bit fingerprint, an ideal value of 37 bits must be different from one fingerprint to another. In this case, we achieved an average of 35.33 bits that were different from one fingerprint to another with an upper and lower limit of 35.62 and 35.09. Similarly for the intra-chip HD, we achieved an average value of 1.05% with an upper and lower bound of 1.33% and 0.84%.

Ideally for each power cycle there should be no change in the bits; our values show that there was a small change of 0.64 bits out of 74 bits changing overtime, with the upper and lower bounds of 0.81 to 0.51.

Table 4. 1 Overall analysis of inter-chip and intra-chip HD of a 74-bit signature at seventy different locations for each FPGA.

FPGA	**Inter-chip HD (Ideal 50%)**	**Average bits different**	**Intra-chip HD (Ideal 0%)**	**Average bits changing**
1	49.32	35.47	1.32	0.80
2	49.52	35.61	1.33	0.81
3	48.85	35.13	1.04	0.63
4	48.82	35.11	1.03	0.62
5	49.44	35.55	0.99	0.60
6	48.84	35.12	0.98	0.60
7	48.98	35.23	0.84	0.51
8	49.53	35.62	1.04	0.63
9	48.80	35.09	0.92	0.55
10	49.13	35.33	1.03	0.62
Average	49.12	35.33	1.05	0.64
Max	49.53	35.62	1.33	0.81
Min	48.80	35.09	0.84	0.51

Table 4.2 shows the stable values on each FPGA. An average of 97.08% of the values are stable. In our case, 5028.8 values out of 5180 are stable. For all ten FPGAs, over 96% of the values were stable.

Table 4. 2 Overall analysis of inter-chip and intra-chip HD of a 74-bit signature at seventy different locations for each FPGA.

FPGA	Stable values / 5180	Stable values percentage
1	4984	96.22%
2	4990	96.33%
3	5034	97.18%
4	5033	97.16%
5	5040	97.30%
6	5038	97.26%
7	5056	97.61%
8	5033	97.16%
9	5048	97.45%
10	5032	97.14%
Average	5028.8	97.08%
Max	5056	97.61%
Min	4984	96.22%

We also took each of these 74-bits and analyzed how different these are from one FPGA to another. Each of these 74-LUT locations can be used as a separate *.bit file as a

challenge to generate a 74-bit response. Table 4.3 shows the summary analysis of inter-chip and intra-chip HD for each location between ten FPGAs. For example, at location-1, a 74-bit fingerprint was obtained at each FPGA using the same programming (*.bit) file for ten power cycles. Similarly, another 74-bit fingerprint was obtained at location-2 from each FPGA for ten power cycles. To be clear, LUTs that are used to obtain fingerprints in one location are different from those in another location. The average inter-chip HD is 49% with an upper and lower bound of 51.38% and 46.15%. For a 74-bit fingerprint, an ideal value of 37 bits must be different from one fingerprint to another. In this case, we achieved an average of 29.67 bits which were different from one fingerprint to another with an upper and lower limit of 31.11 and 27.94. Similarly for intra-chip HD, we achieved an average value of 1.05% with an upper and lower bound of 1.80% and 0.49%. Ideally for each power cycle there should be no change in the bits; our values show that there was a very small change of 0.64 bits out of 74 bits that changed overtime, with the upper and lower bounds of 1.09 to 0.30.

Table 4. 3 Inter-chip and intra-chip HD of a 74-bit signature at seventy different locations on ten FPGAs.

Location	Inter-chip HD (Ideal 50%)	Average bits different	Intra-chip HD (Ideal 0%)	Average bits changing
1	48.39	29.30	1.38	0.84
2	47.49	28.75	1.32	0.80
3	48.95	29.64	1.42	0.86
4	49.87	30.19	1.02	0.62
5	49.93	30.23	1.14	0.69

6	49.95	30.24	0.78	0.47
7	49.64	30.05	1.44	0.87
8	49.43	29.93	1.20	0.73
9	48.47	29.35	1.05	0.64
10	48.28	29.23	0.77	0.47
11	50.09	30.33	1.14	0.69
12	47.92	29.01	1.05	0.64
13	47.46	28.74	0.94	0.57
14	46.15	27.94	1.02	0.62
15	49.00	29.67	0.75	0.45
16	49.13	29.74	0.51	0.31
17	50.47	30.56	1.21	0.73
18	48.88	29.60	0.74	0.45
19	48.08	29.11	1.24	0.75
20	48.18	29.17	1.02	0.62
21	48.61	29.43	0.65	0.40
22	48.92	29.62	0.95	0.57
23	50.32	30.47	1.00	0.61
24	49.56	30.01	1.25	0.76
25	48.61	29.43	1.03	0.63
26	49.01	29.67	1.01	0.61
27	49.96	30.25	0.96	0.58
28	50.50	30.57	0.73	0.45

29	49.59	30.02	1.15	0.69
30	47.24	28.60	0.91	0.55
31	49.42	29.92	1.12	0.68
32	49.42	29.92	0.59	0.36
33	47.65	28.85	1.18	0.71
34	48.67	29.47	1.00	0.61
35	48.70	29.48	1.39	0.84
36	48.19	29.17	1.12	0.68
37	49.63	30.05	1.37	0.83
38	50.33	30.47	1.37	0.83
39	49.03	29.69	0.91	0.55
40	51.38	31.11	0.69	0.42
41	49.18	29.78	0.95	0.57
42	47.60	28.82	1.05	0.63
43	49.50	29.97	1.15	0.69
44	49.94	30.23	0.85	0.51
45	48.86	29.58	1.30	0.79
46	47.68	28.87	1.14	0.69
47	48.67	29.47	0.84	0.51
48	49.35	29.88	1.12	0.68
49	48.23	29.20	0.88	0.53
50	47.73	28.90	1.05	0.63
51	50.14	30.35	0.81	0.49

52	50.22	30.40	0.98	0.59
53	48.95	29.64	0.80	0.48
54	47.67	28.86	1.11	0.67
55	48.70	29.49	1.15	0.70
56	48.60	29.42	1.10	0.67
57	47.93	29.02	1.80	1.09
58	49.36	29.89	0.49	0.30
59	49.67	30.07	1.53	0.93
60	49.05	29.70	0.95	0.57
61	48.44	29.33	1.39	0.84
62	48.49	29.36	1.19	0.72
63	50.16	30.37	1.21	0.73
64	49.20	29.79	1.38	0.83
65	48.96	29.64	0.88	0.53
66	49.00	29.67	0.77	0.47
67	49.55	30.00	1.03	0.63
68	50.25	30.43	1.17	0.71
69	49.71	30.09	0.69	0.42
70	48.46	29.34	1.27	0.77
Average	49.00	29.67	1.05	0.64
Max	51.38	31.11	1.80	1.09
Min	46.15	27.94	0.49	0.30

4.2.1 Fingerprinting

In order to create fingerprints for an FPGA, we took ten power cycles of SUVs. Each instance is called a latent fingerprint. A known fingerprint is created by averaging across these SUVs and rounding them to logic '1' and logic '0' as shown in equation 4.1. For a particular memory cell, its known fingerprint value is '1' if the average probability (p) is greater than 0.5 – or else its '0'. Table 4.4 shows the fingerprints in a hexadecimal format. In this table we are showing seventy fingerprints from one FPGA. In the Appendix section we provided the fingerprints for the rest of the nine FPGAs.

$$Memory\ cell\ Fingerprint = \begin{cases} 1, & if\ p > 0.5 \\ 0, & if\ p \leq 0.5 \end{cases} \qquad (4.1)$$

Table 4. 4 Seventy fingerprints on FPGA1

Location	Fingerprint		Location	Fingerprint
1	2F7FD523174CD8FFDE9		36	35534BF1FFAAE121B11
2	2B7A9BA724BDB5987E5		37	1AFB67AD948C26FE5BE
3	2EF5D3B1F78E07CC9A4		38	27F0C75CBE762FEC2FD
4	165EABF2BD82EE312BB		39	2E434D48BBE0F63E626
5	26C32D66744B3E4CC26		40	00A3D77CA4E9BA7A7E0
6	0FB3F517152EE60EAD0		41	294A04DDBC046107AF9
7	272EB7B76D563DE549B		42	39CA31D4FA7D264A5FF
8	0DBBF1DDEF1F4ACA73B		43	2B6F0DDD2FD4482B2F3
9	1B3AC76D84796484DCF		44	179DE1DA4EF95C79410
10	3FAE4ABBB88CA38CBAB		45	267AB7902C5F29F26CE
11	08EB4F85EF16C792300		46	1A081B726571B526AC1
12	2FEFCF0E97AEF6B49AD		47	264C79BC7BACBCE60FE
13	026D9197B9C957F51CD		48	191B20B0F2BD9327750
14	1947045765D64BD3BBF		49	065C366B88E99C985EB
15	0A22184472DFD83E630		50	36EC4AA9AFFEBFD8EAE
16	31F3B3B79333E7A3A47		51	20E85DBE910DCF3EDE7
17	26600DF50E39A490E3B		52	26B70D92A53ED8B2DAC
18	24685539D65C5AC13AF		53	0916502FC921CB5FB44
19	3AB7E835BCAFC5AC7EF		54	09E46BA7E83398906DF
20	3203BD76C4D2EF381A1		55	245F9E0359FD485F38C
21	2FD9BEB46C4775315DB		56	377FF54663E587C28FB
22	2E037BB28D9E4283370		57	3F2D794B63ECD9D983F

23	11DFCA5E6D7D119DBA3		58	396FCAD6D304C5BD76D
24	2EFBE714E36FADBB38F		59	2CA9FDFD858CD4BD1B4
25	3AB90D13E82DF0F8D42		60	248D7A6884C27451DB9
26	0A63135E531C2ADEF3E		61	27C615EDD09556CA767
27	03FE3705BF377618FD4		62	29AF5F8FBA8CF2B6BC1
28	39BF41ADEE4B52FF14F		63	1CF94C257B60A9F3692
29	264DBF0C105CC6D7B6A		64	2275AFE44A869680FB1
30	330FEC46B4B7EDE459F		65	03A7EFD081CDAA03F66
31	210B7AC6675728A783A		66	2EF103017DA9C2B5E9F
32	28734BCFA907375E1AA		67	1F07AEEEC602D7F7FE4
33	19077FE017F673D22D3		68	38C7F473FF0B0E80B7B
34	087908C1C1EE4EB744B		69	2CFC8AA07314159FE85
35	1B5A7F0571BE30AAADF		70	33C57BBCE3A42F7AAD4

4.2.2 Strong PUF Analysis

As we discussed earlier, a week PUF has a limited number of (C_i, R_{C_i}) pairs and a strong PUF has numerous (C_i, R_{C_i}) pairs. To test the strong PUF hypothesis, we selected a million (1,000,000) (C_i, R_{C_i}) pairs using a pseudo-random number generator (PRNG) without repeating elements. We used the Matlab *randperm()* function to generate these million (C_i, R_{C_i}) pairs from the database of SUVs that we had already generated. We performed a similar fingerprinting analysis as in Table 4.3, but instead of 70 locations, we compared a million different combinations.

Table 4.5 shows the average inter-chip and intra-chip HD of different 74-bit signatures taken using a million combinations of different (C_i, R_{C_i}) pairs. We were able to achieve an average of 48.99% inter-chip HD from these million (C_i, R_{C_i}) pairs. In Table 4.5, an average inter-chip HD value for each fingerprint is in between 43.33% and 52.83%. In other words, an average of 29.66 bits out of 74 bits are different with an upper and lower bound of 31.98 and 26.23 bits. Similarly, we were able to achieve an average

of 1.05% intra-chip HD to an ideal 0%. An average of 0.63 bits out of 74-bits are changing overtime with an upper and lower bound of 1.48 and 0.11 bits.

Table 4. 5 Average inter-chip and intra-chip HD of different 74-bit signatures simulated using a million (C_i, R_{C_i}) pairs on ten FPGAs.

Million 74-bit (C_i, R_{C_i}) pairs	**Inter-chip HD (Ideal 50%)**	**Average bits different**	**Intra-chip HD (Ideal 0%)**	**Average bits changing**
Average	48.9956	29.6646	1.0514	0.6366
Max	52.8348	31.9891	2.4565	1.4873
Min	43.3333	26.2364	0.1862	0.1127

Figure 4.4 shows the average HD percentage of these million (C_i, R_{C_i}) pairs simulated from ten different FPGAs for ten power cycles. The x-axis shows these million values, and the y-axis shows the HD percentage. Figure 4.5 shows the average HD values of these million (C_i, R_{C_i}) pairs simulated from ten different FPGAs for ten power cycles. The x-axis shows these million values and the y-axis shows the HD values for a 74-bit fingerprint.

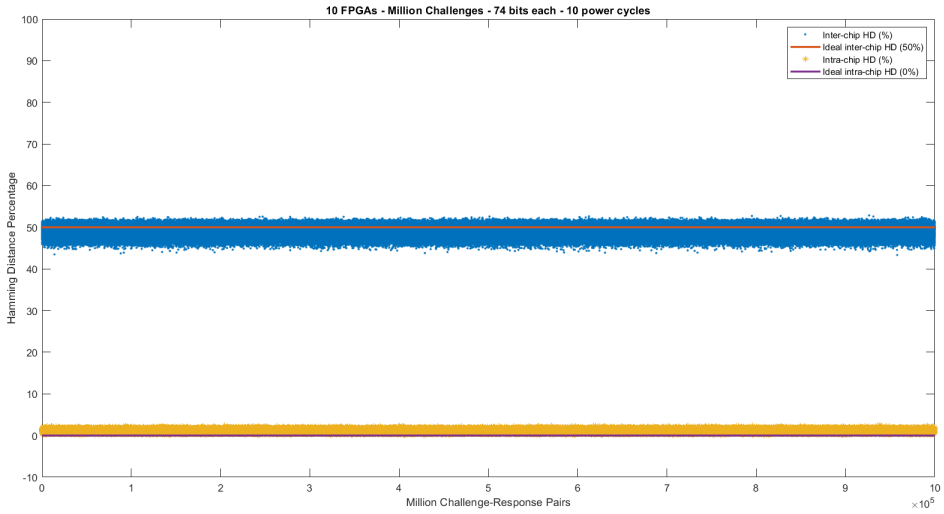


Figure 4. 4 Average HD percentage of a million challenge-response pairs from ten FPGAs

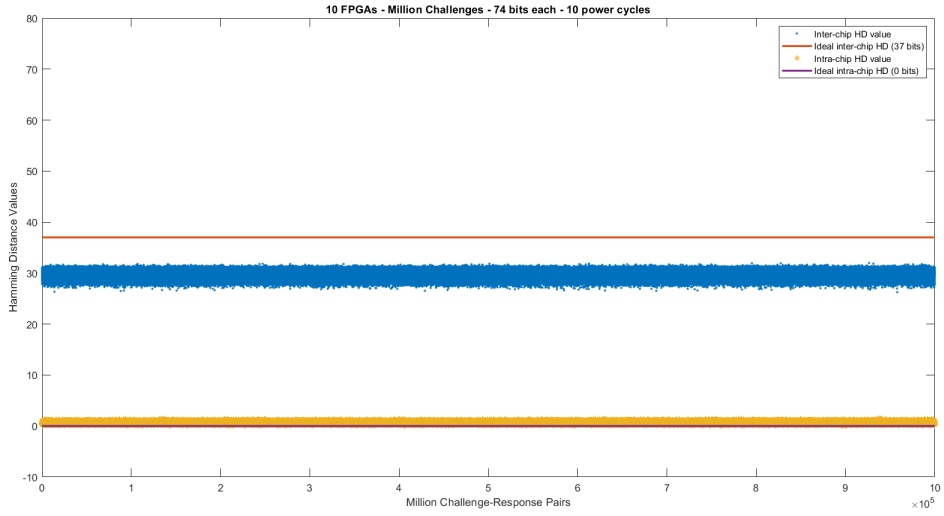


Figure 4. 5 Average HD value of a million challenge-response 74-bit pairs from ten FPGAs

4.2.3 Memometer Memory PUF vs. Butterfly PUF

A Butterfly PUF (BPUF) is another memory-based PUF invented to overcome the manufacturing memory preset [34]. S. Kumar et al. mentioned that "Implementing a cross-coupled element using a combinational logic on an FPGA is not straightforward due to inability to create combinational loops." We were able to accomplish this goal using our LUT-based memory PUF. Our design uses a cross-coupled combinational logic by creating a combinational loop using LUTs. It is definitely not a straight-forward approach because the routing paths are not easily accessible to the FPGA designer. However, with many experiments, we were able to match the feed-back path delay, thus providing us with the SUVs needed for fingerprinting.

Figure 4.6 shows the design and construction of a BPUF. They have used FPGA latches to construct a cross-coupled combinational loop. The preset (PRE) signal sets the output high and the clear (CLR) signal sets the output low. For a BPUF, the PRE of one latch and the CLR of another latch is set to low. The excite signal is connected to PRE of latch 1 and the CLR of latch 2. They were able to simulate combinational logic by setting CLK to high. A PUF operation starts when the excite signal is set to high; then the BPUF goes to an unstable state. After a few clock cycles when the excite signal is brought low, the BPUF will settle to either logic '0' or logic '1' at the OUT signal. Due to the symmetric layout construction, the OUT value is determined purely based on the intrinsic characteristics.

The fundamental design difference between the Memometer memory PUF and the BPUF is that we use LUTs while the BPUF uses latches. Both of these designs are aiming to simulate cross-coupled memory cells to come up with SUVs. Our construction

is much simpler than a BPUF because the LUTs are the most widely available component in any FPGA. Not all FPGAs offer the latch layout structure that the BPUF requires. Furthermore, there is flexibility of routing by using LUTs instead of latches. Our symmetric layout was nearly perfect with 0 pico seconds (ps) delay between the feedback paths. For a BPUF, the clock must be laid out in a way that the skew must be nearly zero for unbiased SUVs. Our implementation does not require a clock. As soon as the FPGA is powered-on we get the SUVs.

The most important difference between the BPUF and our design is that the BPUF has a limited number of (C_i, R_{C_i}) pairs, whereas the Memometer memory PUF has hundreds of (C_i, R_{C_i}) pairs making it a strong PUF based methodology.

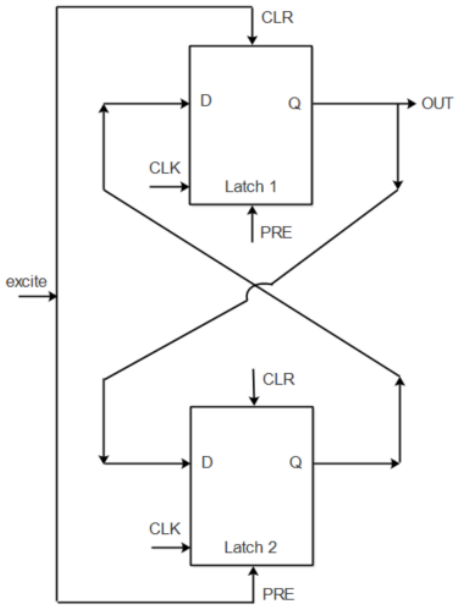


Figure 4. 6 Butterfly PUF design [34]

4.3 Artificial Aging Analysis

As mentioned earlier, we want to predict how these memory SUVs change as the IC ages over time. For our aging experiment we used five Xilinx Zedboards, which have a Zynq 7000 processor with Artix-7 FPGA fabric. The processor's operating temperature is in between 0ºC to 85ºC [62, 61]. In order to artificially age these circuit boards, they are placed in a temperature-controlled chamber at a desired temperature and voltage. As mentioned earlier, we are using the acceleration factor AF in equation 3.1. The parameters that we are using for this test are based on a similar study [38]: the gate voltage exponent (α) = 3.5, the time exponent (n) = 0.25, the activation energy (E_{aa}) = -0.02eV, Boltzmann's constant (k) = 8.62 x 10^{-5} eV/K, the nominal voltage ($V_{nominal}$) = 1.8V, the nominal temperature ($T_{nominal}$) = 23ºC, the higher stress voltage (V_{stress}) = 2.5V, and the higher stress temperature (T_{stress}) = 80ºC. After applying these parameters, we get the aging factor AF = 163.99, which means one hour of accelerated aging gives us 163.99 hours of aging, which is approximately one week. We apply this AF to our devices by placing them in a temperature-controlled chamber for 255 hours, which gives us approximately five years of artificial aging. For a more realistic analysis, we programmed these FPGA LUTs during the aging process. These circuit boards were taken out from the temperature-controlled chamber for every 1, 2, 4, 8, 16, 32, 64, and 128 hours to measure the fingerprints at nominal conditions. For each aging cycle, we acquired a set of ten SUVs. We used a home conventional oven for our experiments and our temperature-controlled chamber was set to 88ºC ± 11ºC.

We used five Xilinx boards to perform five years of artificial aging experimentation. Figure 4.7 shows the degradation plot of the number of stable bits as the

IC ages. Table 4.6 shows those stable bit values as the IC ages. As the IC ages, all these SUVs for the five FPGAs are consistently degrading. Figure 4.8 shows the average percentage of the stable bits as the IC ages. The average number of stable bits for five FPGAs at week-0 is 96.84%, which is 5016.31 stable bits. After five years of artificial ageing, which is 255 weeks, the average number of stable bits is dropped to 60.336%, which is 3125.405 stable bits. The average standard deviation of these stable values for all five years of the artificial aging process is 4.279%. Over the span of five years of FPGA aging, on an average, if we take 128 SUVs, in the worst case there should be at least 64 stable bits to be used as a fingerprint.

Table 4. 6 Artificial aging analysis on five FPGAs for five years.

Artificial Age (weeks)	**Number of Stable bits to a total of 5180**				
	FPGA1	**FPGA2**	**FPGA3**	**FPGA4**	**FPGA5**
0	5032	5038	5040	4990	4984
1	4755	4626	4557	4371	4465
3	4596	4549	4373	4072	4231
5	4457	4439	4207	3863	4005
7	4251	4189	3984	3596	3711
15	4061	3982	3808	3411	3523
31	3757	3747	3624	3182	3303
63	3532	3466	3463	3000	3153
127	3336	3351	3435	2879	3034
191	3265	3269	3377	2777	2939
255	5032	5038	5040	4990	4984

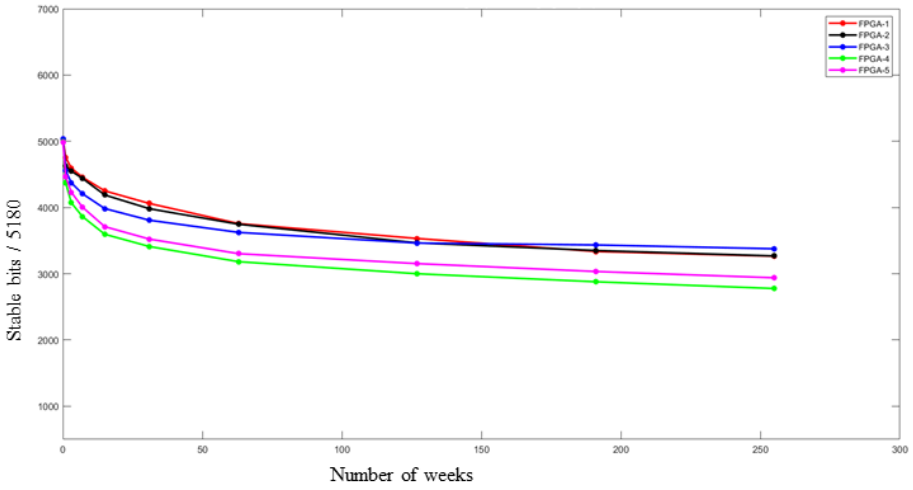


Figure 4. 7 Artificial aging experiment: number of stable SUVs as the IC ages

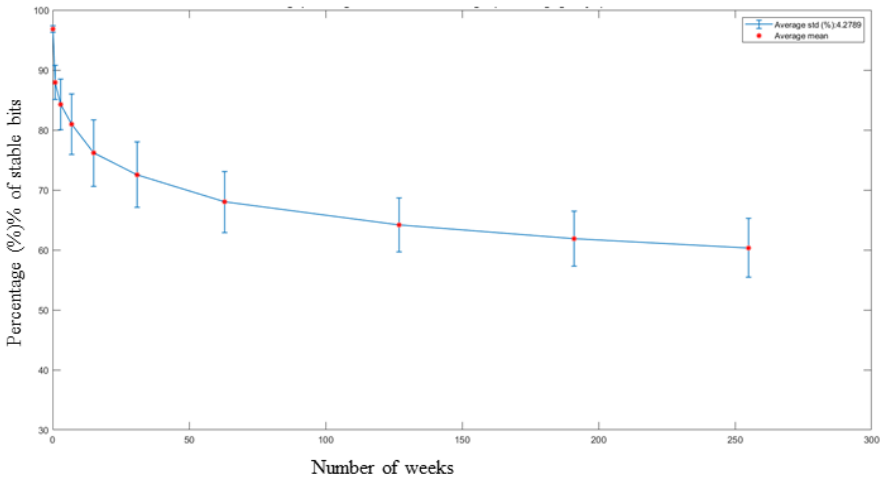


Figure 4. 8 Artificial aging experiment: average percentage of stable bits as the IC ages

4.4 On-chip Analysis

Figure 4.9 shows the Memometer block diagram using Xilinx Zynq SoC. The Memometer Memory PUF is implemented on the programmable logic (PL) side using cross-coupled NAND gates mapped to cross-coupled LUTs. When the device is powered on, these SUVs are sent to the processing side (PS). Fingerprints are stored in a database (DB) on the PS. Hamming distance comparison is performed on these SUVs with the known fingerprints in the database. If the HD meets a certain lower threshold, then the FPGA is a valid FPGA. If not, then it is considered suspect.

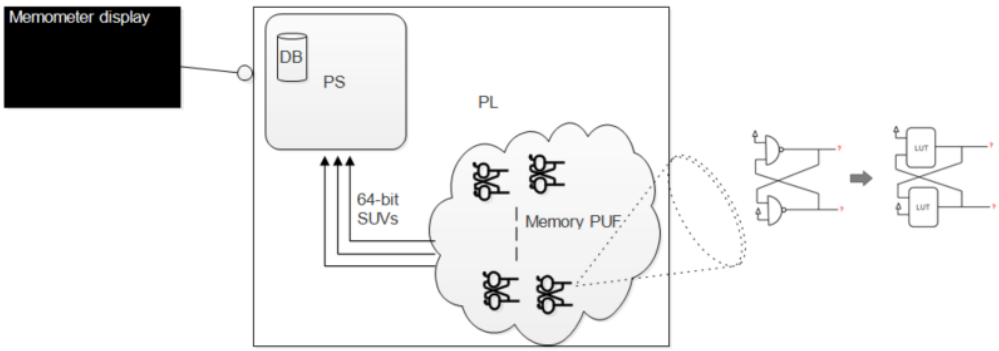


Figure 4. 9 Memometer Memory PUF implementation on Xilinx Zynq SoC

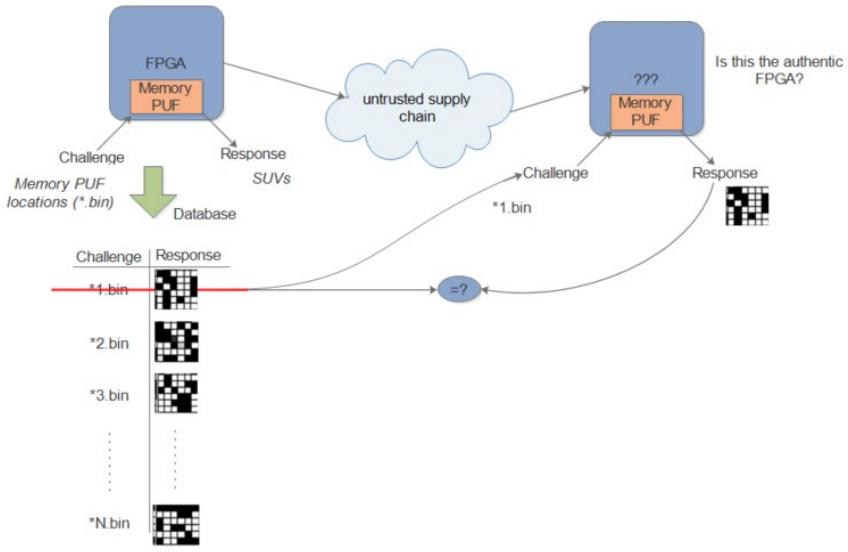


Figure 4. 10 Memometer proof-of-concept authentication process

Figure 4.10 shows the Memometer authentication test on Xilinx FPGAs. To test the proof of concept, we stored a few FPGA fingerprints in the database. When an FPGA is picked up, one *.bin is randomly selected and loaded on to the FPGA. The memory SUV values are generated and compared against the fingerprints that are already stored in the database. Figure 4.11 shows the correct matching of the picked-up FPGA, and Figure 4.12 shows the unauthorized FPGA.

Figure 4. 11 On-chip analysis of the Memometer software determining correct IC

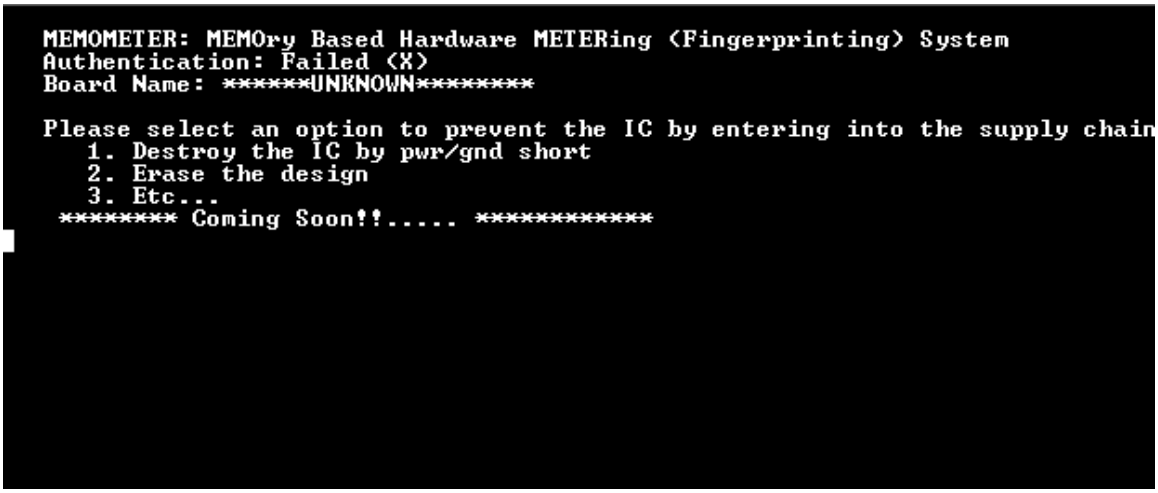


Figure 4. 12 On-chip analysis of the Memometer software determining unregistered IC

4.5 Applications

PUFs are gaining a lot of attraction due to its low-overhead, inexpensive implementation. PUF fingerprints are extremely hard-to-forge and unique to all ICs, ASICs, FPGAs, PLDs, etc. Here are some of the applications: IC identification and authentication [53], hardware obfuscation [57], cryptographic applications [23], secure processor design [54], True Random Number Generator (TRNG) [19], locking/unlocking ICs post fabrication [31], etc.

4.5 Conclusion

The memory PUF-based metering system called “Memometer” is proposed to address the hardware security and trust problem of IC counterfeiting, cloning, overproduction, and IP piracy. A generic HDL kernel is developed to access the IEEE 1149.x JTAG TAP controller to read back memory SUVs for fingerprinting. This HDL kernel is simulatable, synthesizable, and extensible. It is compatible with almost all JTAG compliant ICs. This makes it completely independent of FPGA and ASIC vendors as long as they adhere to the JTAG IEEE 1149.x standards. A practical memory PUF-based fingerprinting methodology is demonstrated by developing a cross-coupled LUT based memory element for overcoming the power-on memory preset issue for programmable devices. Our memory start-up values’ inter-chip and intra-chip HDs are very close to the ideal 50% and 0%. Instead of having one fingerprint per device, our methodology creates many (hundreds) of fingerprints per device, basing it on a strong PUF. We include aging analysis in this study to demonstrate the fingerprints’ gradual aging process over the span of five years of artificial aging. We have demonstrated the authentication of FPGAs with our on-chip analysis Memometer tool. Our future work includes the generation of a unique-key identifier to track ICs and the application of error-correction to these fingerprints to withstand harsh environmental conditions. Instead of passively metering ICs, we are working towards actively preventing them from entering into the supply chain.

CHAPTER 5: Future Work

5.1 Error Correction

The Memometer is a supply chain security tool, primarily used for tracking ICs using unclonable fingerprints based on memory SUVs. Our future work includes developing a methodology that can generate a unique key identifier and be able to track ICs within the supply chain life cycle, perhaps taking advantage of cloud-based infrastructure for storing the keys in a secure database.

Also, we are developing a methodology that leverages and adds error-correction to these fingerprints, as these fingerprints are subjected to IC aging and exposure to adverse environmental conditions including extreme heat, cold, radiation, etc.

5.2 Active Metering

Instead of passively monitoring the ICs, we want to develop a methodology for actively controlling the genuine ICs and preventing the untrusted ICs from entering the supply chain.

Most ICs these days come with HW/SW co-design capabilities. The fingerprints that are generated at the HW side could be used by the SW as an authentication password.

Appendix

The purpose of this section is to show the rest of the fingerprints from nine other FPGAs. These are seventy fingerprints taken from seventy different locations on the same FPGA. These fingerprints are displayed in hexadecimal format (Tables A. 1- A. 9).

Table A. 1 Seventy Fingerprints on FPGA2

Location	Fingerprint		Location	Fingerprint
1	240B4B6F572BD334193		36	3F869A7F984E3CFE279
2	3E5E95EF652FAB40F89		37	2A708E1DA414C96D69F
3	35AAF40547C0963FC53		38	1E26C644F1949608D81
4	0434658E347BCEEF4A3		39	171D362B47431F0CACE
5	0D5C0B02867EDC3BD5C		40	3CC471917698325D178
6	1A711945BFE414AFEB2		41	27A4F7A64B41FA95F15
7	19C93E8DFFD1F0D19F5		42	263E1C3A7BFC9396D1D
8	13BC23F5B5D98CADBCF		43	1770CE52809E50AABB0
9	2ECB3E307951AFAC0B0		44	1B347F358B703DEB283
10	36EDE3918361EE5AFA7		45	09A143778FBF4D73294
11	3D3CE2DFBFEBCBBD256		46	3A76E724DA8E31B3A49
12	1EC91DDE7376B3A5EBF		47	17BC1975EC5DD56EABD
13	0F3565B9BEFE71BD64D		48	0FB6CA18F0359A46754
14	1FC9D546D20BACA9DBF		49	31CF50133CE49DA9517
15	232B17B9A9F5AF2C139		50	3ED63CF1DBE6EB9BFAC
16	39EED7D4642F46F32B3		51	39CA76937F4C9529B0F
17	22F40B75EC45F8E6BC7		52	3FAF8817F20BFB3FBF3
18	3EF3BB6BB39B258A98C		53	3F917493D7A003518BD
19	11D93208CB8A24D2100		54	2AEEE879D06D074FFD1
20	1DEA765E306ADBFE412		55	22A393CFB242CDD2375
21	3DCD1EC8A1AB2598B53		56	11A27771FFA329CFC7D
22	3E92164045CFBCA9C23		57	167E004E7E9AEB670A0
23	03B3337EFC72905A80A		58	20E6B2C7107E529C635
24	059B91FE909A8B03BE8		59	30E3426C554A34E73C6
25	06E27356CD3A0BBFF85		60	12DCA18BDBA3C469BA8
26	07FABBF3EE9A7AED1FF		61	33B14792A47EDA7CFBA
27	1D1AA6B0C7CA21D21D6		62	395F5E118DCA3E4D167
28	1B94EFAB650B174A028		63	3F0BD7AD91B0DF46F8F
29	14EE45E18E88D7E5B12		64	1FB3529D5C6D4CD4563
30	345FE2E1DABAF5A616D		65	3E35BA6649EDB3AB874

31	1E7A02DBCB463CAB7CD		66	36BDB7753CB37D9351F
32	1D9D63DC6FB4FF6BD21		67	22AA2A52C7B478026D2
33	229AAE9F7912BFCBDBF		68	0C864D21BF49596EFA2
34	229EA2D9B9B16583007		69	2F741AB0EB762C29FB5
35	39EB53C953097E397DE		70	13FEDBC98ACFF4CFCDC

Table A. 2 Seventy Fingerprints on FPGA3

Location	Fingerprint		Location	Fingerprint
1	39B4EDFE36338FD486C		36	3CF526B70EEF3F4AA9E
2	319EA9CD5968919EC70		37	2A867BFB7A3207778BD
3	397B49F54534E3EDDC9		38	14EC86F1380C2B9F243
4	1651B6C7927639E36FD		39	0320BFF348DE5B3D9CC
5	2C37C75A477CDA17179		40	39C78241A1AF0D4D11F
6	2FC7A78F7964B3A6B54		41	1DCE2A77FDFB749B37E
7	1E0FCF2AFBD47914C4A		42	3F574CF916BE365CDED
8	295F978EFB77DA3F5EA		43	366B39C3B4FDBD1F95B
9	3B863152E869FED556C		44	185EFB700D5963DBC72
10	1CCEF13D604EF13101D		45	2830C9FC34BF6F56E48
11	295BD83CFF673F3489E		46	3F5FFE89F395708EA57
12	1F9A8FFCAE15B184E7D		47	21762BF88D38E52B27F
13	16BEBA3BF76AAFB55FA		48	0D8EF34E13F38395790
14	062FD27F0F77237FD6C		49	0F754F5ABDE59F248BF
15	07FD38CADA0EC3FF3C8		50	393EECC3AFDFFFAFD93
16	2938D714EFDD6DF66B0		51	0D4553DA8C1FBFC8A87
17	0B97053C919A9D1E8A1		52	3402AFD4E73B79295D9
18	0BBED0ABA4DFD0F6E04		53	1D46F4CCFEA78680591
19	2DA1B6AF99C0D351DBA		54	3CDF10DB001D178A2DF
20	2F99A1B3F11CFB3EED2		55	00AB4AE789AED93CD27
21	1F21C7C55F5FFF5F5D0		56	3511954A3EE49DAFBE7
22	20EEB6B7A7FE66CAA63		57	0F9DCDB7A518D6E3CFF
23	1C5B57ACD7717ACF79D		58	38B9FE7739FAADBFC5F
24	35B8C0A4CC0D5BB2F7B		59	1FDF73E17A68E9FEA7A
25	3DF289D9C5988FAC22A		60	333ECE5D17ABEBFB62E
26	15FE281D3A0E01C4D3F		61	126BD19946DD2F37961
27	2AB31945F9CB6C3FBF9		62	314971CD2A6E6BAF970
28	056FDC613F6A3B3D39B		63	0CE5BB021331749EFE1
29	3DBF80AF2FEFDFCDCE0		64	118DA9B628893F6E3F7
30	251487842FCFF8FC77F		65	1F3FADEBBDF7F7AA532
31	2A758EFA9C8B8BFF3DB		66	29AA495AF6ECF6D9F7B
32	14DB6FF502EC056CF7A		67	2D146ED03B284E7A4B4
33	241CF3237A2A973F6C3		68	0279B99FF9E15D53E63

34	19CFEF9ADD67E9C3ABA		69	25CACD404177F3BE173
35	33DDD6D223EFE5F2EA6		70	3EBE71DEB166DFBE6F1

Table A. 3 Seventy Fingerprints on FPGA4

Location	Fingerprint		Location	Fingerprint
1	185AC02431A0F397373		36	3A35AB27A31D7EB5AB8
2	3DFEA3072F9CC866CFC		37	3F1BB0630EAC7460C96
3	13F8BECF9D2F46F83E3		38	37732FE4BDD53A9C8FF
4	1FD77C1A6D6BED51D5D		39	246BBDDD65DE1EFD6A4
5	39AB76EF56D7CBDDB33		40	1720D92529CAD353D25
6	0079AFA11B7D133ADE0		41	358E8379B2D30991CD9
7	15B2CE9B3FDE6FF82C9		42	15EF7EED1E8DE5F8FFA
8	2AE07FF8FD93FDE7255		43	21176FCF9253DFFF260
9	0B21C72B2B0DFC066FE		44	28C7BC1D67E56BA779F
10	237CB2B9F13FEB4FF5D		45	1C7B9CD0F6DF556DA99
11	20C4C77654F5E2D5165		46	2A6D5E43AAA8B4EFA47
12	18E44192783F6EEFA3E		47	387F463EF18B5EDBD39
13	17D532CE600396EB813		48	107B3CEAD75FA5FF837
14	0C15CCF0E5A1F4C3BFF		49	2F7F8D5709DBE787D73
15	2A94F3D0547724ED807		50	33D237C587DEB5345D7
16	37E58B5FB14B0F6FA77		51	1CD14E394AEBE6EBD53
17	179A6688FD1EE9B4929		52	26CCFAF7101F36D58EF
18	17E42659DCEE6FF69B7		53	1C5FB9FBB9E6B887F0B
19	3BB5EDE773FEFFD7ACA		54	3BDFAAA93867E7DA7F5
20	2ADFB3B6D2E46562392		55	27F5171B1F50EE09C5C
21	3F5459ECF35ED3523EB		56	3DFF7380CC157B6EBC4
22	1CE60B32FB8D5CCBA39		57	3363F3270318DF73F4F
23	1D32B59A7EF67D25E7D		58	1BC493041E75097CAD6
24	31ACAC8F7FFB88B0B1A		59	23CEC7B86A827EABC4E
25	2FAF4D2AAE7EF7E9C56		60	10ED67ADB6B373747EA
26	316652CA1D2CEFD29FF		61	1285C7CF25D73E6FDF2
27	3AC7256B7FFEB82DD6F		62	3C1673298F3558C6AEB
28	135D4FE695C5E896F11		63	239DD682373BC916543
29	1DF6FA9F93C6098D84E		64	229462EFFABE4751A28
30	102E6EADA395FB6B001		65	1227B5A5598FB7845A6
31	35DADE34F667595D709		66	2AD0AEF6F0DDDF0AB44
32	2F7E2EEF76D47FA71DD		67	0147EBF34EA31F22F83
33	35B874B47BBDDCF5FF9		68	30FACDBCF196A7A8537
34	1F4D3DD5AF7CBFDF5F8		69	25895D33DED9BF65CA0
35	1BD7BBF8963DF32676E		70	1BF6FF97F9C66C1077E

Table A. 4 Seventy Fingerprints on FPGA5

Location	Fingerprint		Location	Fingerprint
1	0FA55B5CFB50C57EDD3		36	3F7198F7D4CB36EB713
2	10FBFD668C89493AEC7		37	319ABD892FBF7ACAD0D
3	0FFE8559B37DC4F0A3B		38	2655BCC476397A6C58D
4	2C9FB70202C79BB90DE		39	35BAB48FABAB93F5CEF
5	175DAF2ABD95B2674FE		40	0B9F62DD7FE4B5E3BBF
6	3DB54E12E79DD7EA6C1		41	056ACB6393ED97D1407
7	3BBAE62E565325E75D6		42	2C7FBBDD1FDD8F92149
8	05761F7AA7DD69E8EB4		43	0264AE970F6D0F63FB1
9	0A7F25E96FFEF4D873F		44	10C0BD0530224DD7D9B
10	3C59ADB4CB53DC67CFA		45	10FBC474C66DCED2F41
11	326C07A3EB521597D27		46	2272FEFC9619FF0DF82
12	08C50FB7754ED875FBF		47	21253F36302E65E7B67
13	333F55CE71F6FD897BD		48	2BAE6D6973955CF48F6
14	1FBD36E4ECED5F66FFE		49	1BEFFADFC5B015ACBF3
15	2F37F0D4BB3258FEC82		50	1B7E1BF2612E15FE089
16	37D867E5AAC509DB1E4		51	375D53F41F3BA465296
17	1591DF6F996AE3B1EC6		52	0364A7750F6C3557947
18	2245A477DEF4EBFFD7C		53	2D3112012B03D1A7C9A
19	09A97E93DFF2F6395E7		54	2BE2C56F343D7629D3D
20	270FD03CB8ACEF967B1		55	14E22C5964825C98651
21	05B096688AB09F2C13A		56	17F16D0166C18A2EEEF
22	023E32ACABEB79616EB		57	1A6DB3CCD66DC2FB739
23	37FF2F8C008A9DED353		58	3BBCE0ABDEFAA485CD0
24	27A76E9B4166B7FFAFE		59	28AF423444B8C425753
25	2A355DC7B2146BAAFFE		60	261B33D7AF227BDEDFD
26	1E5F67EFD24600A97F0		61	322D7DF94DF714BA557
27	148030EE7DEEA7FE5B8		62	0ADE7FFC3F7F347F05A
28	16110EC6921FAFB0DBC		63	04F8C098C7DC32D5DCE
29	3016DCBED05C00ACD12		64	324D25C564AFB75E2E3
30	26BAE629B3E72CF4743		65	2E8613056578EE62F74
31	3E942202EEC377F74AB		66	1C7A89E2F3CB54CAE8A
32	0CE0D55D657F7B1502D		67	2DCBFCA4587B96E4E24
33	2C5DF9341F2DDF6FD3C		68	3195BC4766325BD7911
34	3AE4F7335820234BAAF		69	037B57E1F16BCC1CD39
35	0B41B505FCAF90B6706		70	0DC665D9E1233E68C56

Table A. 5 Seventy Fingerprints on FPGA6

Location	Fingerprint		Location	Fingerprint
1	04D77360A6E3EFBBFE6		36	3E85BD251E58FCCF871
2	37EF4AB720AEF8A34BE		37	0BF3DA45691C9B13F97
3	2C5ECBEB1D07BBB32D4		38	0DA736117F3B4AA95A8
4	0C4BFA11BC481776335		39	03BCB5EF05D1DCDAB2E
5	2C2FBDA176359EF62D3		40	1FEAD1A31D97AD1F3EC
6	3BD8DEFFA99F7FDEFC7		41	382E9F97FD9B14FBCB6
7	3E5612740D97655ABA4		42	335F22B42ED8F48F93B
8	0F0FD708F8057685964		43	1F327F9762FC6F9BCFD
9	13CEBFD735CF69529DB		44	287E5D4A7837AECFFFD
10	08E86DE589FEE3BBF5C		45	193792097CE2FD905BC
11	3352D4D42BCE26BB1B3		46	1EFF317C1F7AEDE8431
12	0FD6F677A4568E0BC47		47	0B6CF480FAFB0DCCF0D
13	2CDFDEBD321C1383AF7		48	13432CDBF9ABBE752AE
14	1BFED1DAED6EC6E59F6		49	3D6E31834A886AF729C
15	1223F7719B58DDDF4FD		50	0FB61249CF2279AC9A8
16	16C09DC57BD4AC8906A		51	10FF65B87AFDD3AE603
17	3273885AC7591D4605B		52	1EEB9356C49FF6872EA
18	1E46457307B8FDC3F25		53	010FE85C1168AFA7F7B
19	10EBFFAED219B3F6F3F		54	086F497E675FCEDDEFB
20	2B72B29ECD1703F6D6B		55	26C92055B304ED7FF53
21	333BBE09C6730BDF916		56	278E6FDB71757C13A15
22	3EFF216AD4FADE19D2E		57	35FCFEF67FA0529BB24
23	3FC9A4F7CBB517DA7CB		58	2D0B49EDB9FEEFBFD20
24	36E599C6F6D6AABC854		59	2A0ED8927AD8ABA32F9
25	29EDCF977AEFCC669F5		60	3B346BF35A48E5FFF38
26	241CF32025FB59AB696		61	2DCBBF3D9BD75F93A6B
27	347DD6AD4E6E10E80DD		62	3C3E2DFA3DE0FA5CFCF
28	3D0AA3D7F4F6F62FF99		63	217E7B9CCD9ED69F4BA
29	36C975D919C2DDBBE2D		64	2E4C3B7D8F8F11C63FC
30	25A9D93D978EB4EEAD7		65	0DAAD4776A7F72DA1A7
31	26237E19B47BEDFE5D8		66	29A28EFFC36CE24846F
32	23063F0F6CB1E11F078		67	3E6A530FF1FF77FC38F
33	21CBB8ABFEFEDDFFBF5		68	1D67BB270525DBF6FB3
34	06FEB76CE346438BEF5		69	1BC5327FD3612B83EE7
35	158B6EB4D95BFB15DF8		70	042981B681E938DE3D1

Table A. 6 Seventy Fingerprints on FPGA7

Location	Fingerprint		Location	Fingerprint
1	2D1EB24F3DA8BC2FD25		36	38CF37D65B37D075D5D
2	1BEBAE49DDDB8EF5C89		37	16F941E305D1DD988F6
3	317EAD894AE54BE48AC		38	2DDAFE2F3A535C51FE9
4	2CA78E645A176E73AD9		39	2BD5ED13EC35BFAEFB6
5	1F0DD9F73915C00F73F		40	0F4FBCE492D0DBD37D2
6	27A1BCE0D8DB08F643F		41	19A1BBF3F637D53AEF3
7	3F7D54AE6716C63C343		42	315DF0F3A459DBFAE74
8	2C9A0A763B87A5C34EA		43	375CADAD7BB1F34533D
9	381BD90C0DF3F83F0E9		44	3767D8EB9BFB50AE302
10	2A9EAF2741AFAA6F67B		45	2C9FE77C1F8E79BE7FC
11	0F5EA06CB66F92A9D9B		46	2A767E8F72B5A6E78F3
12	2FF3B110672FFD55BA2		47	3E74572AD1D594E5129
13	2657F9DB374BCEBEACE		48	3CB83A27DBA9B7526DD
14	17BDF9FA3FCC057EC6F		49	0079EBDD596757DEDF5
15	1B7AFE9279227D8287D		50	1DE986CE19723F76573
16	1BCA1CD1B90C14FD940		51	07B9B4FD3920E54C4DC
17	0E3AC30CB6C963C7ED1		52	329DAAAD2DB41A13AC9
18	1E99F578E0A7DCC2B8B		53	159C55B3E6895F7C78E
19	1A7EF7968C9A15D1E09		54	25F2F62773F3D480A53
20	19FB59ED725416B2676		55	32AAA7B2F5E8E3B6A30
21	3C9CB7C6406BB923CDC		56	169FB7BA47F60E96FE5
22	357583BF78E1142FAF0		57	3733B32366D7B76F2B0
23	29475FFA1EF3DCC9BB0		58	1F4E96F63743B9F8A37
24	0F0B857BF7E91F29116		59	1B7361B016CC8B3DEAA
25	2DD7737F8B6DED7BDE5		60	36C7EA7E2B61E3D734E
26	07F3D0DC39A0B3BBF64		61	0AA4E54C6DA0DED3B72
27	2A18FAFFD2634B6B91B		62	2EB438F8E6B1F7DF597
28	185D8C769BA7592399F		63	3EAFAEFFF3EDD86C5AF
29	3D9E1A6C25F6B9BCAC1		64	3FFDA0C77DA9713CBC9
30	1EC5D540DBCCE6B66ED		65	215755058BB5593ECE9
31	2BAF47F53EDBBF952BD		66	0021E1FB2C631F90DB6
32	1E9FD65C0A0FB7C69C3		67	25E3F4ED0ADAF2B9DEF
33	045F7B7B50E77F31BBA		68	2A5518BFB93BADE0A7C
34	31CE9A28B2892C1F7FF		69	0DB01CA79D3932FFF73
35	16F5FDEB5AF00990358		70	002DFBFA27E4EE9D380

Table A. 7 Seventy Fingerprints on FPGA8

Location	Fingerprint		Location	Fingerprint
1	2D9C6B45D77FE988E0E		36	18173C7910DE554A209
2	38F9BFD5C63F75BC394		37	25D2A12CCA991B9DE94
3	3EA5DDC36774E0B9681		38	0D277BF78BEB8C061DF
4	2B7ECD749716FFE2DBC		39	3E6B8B79B017476E51B
5	11753FC7FB6DB7F7E25		40	2F18E6B4D41B4EFE4B1
6	1D4E23E9DE8B5FB7117		41	0033C2AA159426B5CF1
7	314BAFA314AF0AC281F		42	1577A765C27554DD5A5
8	027C6D62FC705BE89E9		43	096E63DABE5E1075307
9	22DD863DAC3585B66D9		44	25793DD864DA9D76A3D
10	36ACFFCEFF6A4DC2A90		45	074BE0EBFC5838CFD7B
11	26532F85E70DBA5F2D8		46	2EB2EDABFD63DACC32F
12	153BA1A696DAD5DE1CE		47	1EF747907C1AFFFFB43
13	1F4FE24BB7DC21B3665		48	3201FF7622DDBE30E4A
14	1F1433D3610ABDFB52F		49	1668A06D9ACD3BB6581
15	0E5150E4FDE09276A37		50	3CBED4CD9E744E1CE2D
16	3011BC2FBCC329D4706		51	271CF7A06CAB63B0FF5
17	35FEC37DA838D8E9718		52	203B5DCF60B18B140BB
18	0F07665D0016D0D16E7		53	1327FFD8F90CF7BFA4D
19	3EF1EBAF85142DC07D1		54	374FF5570E26B4BF784
20	0952B67DA4D0EF3564F		55	013E7E5F1FF25FFD16F
21	1ADF5E4CE4FB72F788D		56	218D38CD9CB03CF7DFE
22	3D026EF4F256A3B9C0A		57	1595BB374753E74F3BD
23	2278BA29FF1704D264E		58	07E92948F52E4D3F647
24	26A5F0C19EC6FC578AD		59	22729FAB76FC9F19781
25	2FC98756EE1AB7921EE		60	2902F7DA63F8B847B5F
26	3E19B0561B5BAFE1AEF		61	3EF921904ABA3FE5B53
27	2F50678E49BABCFF8B4		62	17C868638A35426273F
28	36FDA884A4F3779D985		63	1D23C0A3B00DD2303A7
29	0A7F4EAE39C94F75E60		64	02A753268F72D4C5088
30	39EE58286F9963BCDB3		65	0CA73573C7256315C15
31	1CD207C5B969E3267F9		66	30DD5F5B5B3C96E03F5
32	28DB5CBC2DF56D4C494		67	0AD70539A25357BA6D9
33	2CDCC66CFCCC6FAC3BE		68	3EEF4380C5F517CBAAD
34	39CACF3C61E24AD6D6B		69	15B9B7848F379733CF6
35	0A872EBEC6B9D496C21		70	1BF95F8BB8D51B6833C

Table A. 8 Seventy Fingerprints on FPGA9

Location	Fingerprint		Location	Fingerprint
1	0EDDF0A7A916D50BF3D		36	03D3C5ABD0D6361C969
2	2E7F6D0FE2EB8EAE535		37	2EE6C2DE4F37A9B023D
3	22BE2FBD3BF1FA76DF5		38	32B97B0AFE7905A7A35
4	1FBC7D2DC2E4CF3CB95		39	3E33AE64BAB3D6BDCE3
5	34CFF270DCA46DDEC59		40	3CDC6BB840FFF953C03
6	3CB5DB89453264BADCA		41	0D562A399B02BFFDDAA
7	31FDFBEA6E8999ACC7D		42	1DBC5673FBD56E7E8D9
8	37B42EEA34DC149A197		43	29BA731C0D7BE09B4DD
9	18DD731EED6EE8CDDBE		44	225407822FF3CDD60B6
10	12BD2F3D4EADF6C58FE		45	2EB1AE3DC337A1D5E7C
11	06AADAC2FACAFC834BB		46	1883F15F4FE4B795073
12	1D9715EF3FEC53ADCF1		47	15CC1E4ECEF3C795AE9
13	1325F1CF1F3E9C43AF7		48	1111733D5FFF6A995CF
14	2BE739F60A8F26EF9A8		49	0F778A66BFB8F42E577
15	3E16B75C7FCFFEFD6FC		50	1BE5BB53D6BE743FF1F
16	363C6DFB9987E599374		51	19667DED814128D7EE6
17	113D3AB59BC5BD7A754		52	3191FC91F9C74CCFCBE
18	2F30BCE359D54AE593F		53	3045FF9DD5B08ABE8DF
19	15EFF3D9268EE5AD5AC		54	02A8382CFA771676597
20	1D2681DDCD2C66EACE2		55	30D23B84D1F236FBFCD
21	2BA104C6F6A393EE73E		56	395ED976B793CC14EB4
22	2BA62D1CD737CBFB39F		57	1A1CF8FE1E3AFDFD57A
23	017207CB717D48F1CEC		58	39C3E74275B2F373DE6
24	1BD5BF8CF79254EFB62		59	1ECBCF71EF8604FB9BE
25	1227F76A73FE5039FB6		60	39BF7DD3E0FFA8029E4
26	3744A874531791B5A7F		61	29FD13908AA6AA276DA
27	09E67FFDE1DEF677FBA		62	3EF3756B72FF6E4A7CC
28	1F27D401464979FC442		63	1A8E83FC09B4B516B53
29	0DEBFF3B1F54F6F0377		64	2F08DFDDE18F201F32B
30	3FCFF28FD5FFE8B8D13		65	0BA9F1EAEC690C1D8D5
31	1FD2801CBFCC66782F0		66	3FD8D7F7146CBBB1FD3
32	3F5EC0FF20D2D122715		67	2C6765F11CBE35F617E
33	1307FD7DFA9242F6E3E		68	3F6DF23BE91D7463D6B
34	194EE3F163B4292D04F		69	22B9A3BFBCF6DB7777C
35	25EF9EC3C6FA5E939CB		70	11251D7D1E9BA92E1FB

Table A. 9 Seventy Fingerprints on FPGA10

Location	Fingerprint		Location	Fingerprint
1	197CF87FCBE3DBF6573		36	0ADEEE37C4B11B6EB2A
2	32B3E345282F463DF46		37	07F049DF65F6BE6FDB5
3	17BEDD32E9EFBB7F559		38	3BD2E0DE96E8B67F3F4
4	1D1AEF0EEB9D5D88BCE		39	3AB3E5897D6BC1EE060
5	13BE0FF6D2C1FD9B8E6		40	113DD2EBC82E6F0529C
6	2FCBF7D38AE363BF76F		41	1FE5EC3D135CEF97F3F
7	1EE9CFAEDE2F1B31DB3		42	35858DAFAF3FBDBE621
8	240AF57CB85FBAFFA5A		43	154570E63A71CD52EBC
9	30D69347CD0BFC56E51		44	0F8AEF556FF48BF7B7C
10	12812FF99629DBEDDED		45	03B3DA92BF2DAAC4856
11	37B23D925FA8CF8B316		46	3F2F263B7ECDE5AEA17
12	1FFB0B665FD55F6B6F7		47	008EBA5668E3435A2FD
13	28D7FB5B77632D4636F		48	0A338FBE63EF77C3804
14	2E6FF16D2D735363F3A		49	2047A7FC1EBE710F1D6
15	1A7F6A9DC3B068B987F		50	2FF380BD7C356EEC936
16	3877D06AF5AFB3C53A7		51	2EF11725D7B2EA73CA0
17	0AB4DFE2ECBC99141B1		52	3E9EE4E943E3CE2A8CA
18	14F810C4F36BE6C9EB4		53	3D593FFEA62B3BFFE03
19	2BB0A6C11E26B7F7EC3		54	2E5A7637C42BFCBBF9E
20	0D4B552BBFAC40B8F34		55	303B32F73843373C3DD
21	3988FF587989FDE4C7E		56	31C0FBDEABDCC3D6106
22	2F56F71313CD0CFBA2F		57	37368EDA43E95897552
23	0E08B93379E3F59B40F		58	05FC4E5A721B9CE8C0D
24	35E732D78BD3B78D9EF		59	050FDE2E5C79FB5E103
25	1BF9A0D3DA9ABF6D1FE		60	1BCFC56940CF44F938E
26	0EE9EA187A8671349CF		61	165059DDF7FA39A74B9
27	260FEB8BE3AE6756C41		62	1FB74E6FFFBDEFEAA0A
28	1B28957E418C87AF9A5		63	15227EEEFE2CBBB8DED
29	2D054ED1363B1D41DB0		64	35F428BF948EDDEFE00
30	2071772C7BFB72D2281		65	3D94078E79FFEF5D0DE
31	139E60A64F9FC307CF1		66	2641EB7F1E82C6193DA
32	10A5D153A480BB9C477		67	22225FFEDE33DCBE041
33	07F61F4DDAD13EF969F		68	2B4E4FD6F647BA3D79F
34	1A8DCF2B670DBF98626		69	377C30085A368A7321B
35	052F54827AFAFBB5790		70	35F0F0AE106F5EDF3E0

Bibliography

[1] Semiconductor Industry Association. 2020 STATE OF THE U.S. SEMICONDUCTOR INDUSTRY. 2020. url: semiconductors.org/2020-state-of-the-u-s-semiconductor-industry/.

[2] Bloomberg.com. The Big Hack: How China Used a Tiny Chip to Infiltrate U.S. Companies. 2018. url: bloomberg.com/news/features/2018-10-04/the-big- hack-how- china- used- a- tiny- chip- to- infiltrate- america- s-top-companies.

[3] Ruhr-University Bochum. Critical "Starbleed" vulnerability in FPGA chips identified. 2020. url: news.rub.de/english/press- releases/2020- 04- 16 - it - security - critical - starbleed - vulnerability - fpga - chips - identified.

[4] Mafalda Cortez et al. "Modeling SRAM start-up behavior for Physical Unclonable Functions". In: 2012 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT). 2012, pages 1– 6. doi: 10.1109/DFT.2012.6378190.

[5] Frontier Economics and International Chamber of Commerce. The Economic Impacts of Counterfeiting and Piracy Report prepared for BASCAP and INTA. 2017. url: https://iccwbo.org/content/uploads/sites/3/2017/02/ICCBASCAP-Frontier-report-2016.pdf.

[6] eMemory. eMemory's Neo PUF. url: https://www.ememory.com.tw/en-US/Products/Product?guid=19081314113656.

[7] John M. Emmert and Anvesh Perumalla. “An Asynchronous MPGA THx2 Cell and Architecture for Mitigating Side-Channel Attacks”. In: 2019 IEEE National Aerospace and Electronics Conference (NAECON). 2019, pp. 232– 235. doi: 10.1109/NAECON46414.2019.9057912.

[8] John M. Emmert, Anvesh Perumalla, and Luis Concha. “An Asynchronous FPGA THx2 Programmable Cell for Mitigating Side-Channel Attacks”. In: 2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS). 2020, pp. 840–843. doi: 10.1109/MWSCAS48704.2020.9184563.

[9] Maik Ender, Amir Moradi, and Christof Paar. “The Unpatchable Silicon: A Full Break of the Bitstream Encryption of Xilinx 7-Series FPGAs”. In: 29th USENIX Security Symposium (USENIX Security 20). USENIX Association, Aug. 2020, pp. 1803–1819. isbn: 978-1-939133-17-5. url: https://www. usenix.org/conference/usenixsecurity20/presentation/ender.

[10] Dave Evans. The Internet of Things: How the Next Evolution of the Internet is Changing Everything. 2011. url: https://www.cisco.com/c/dam/en_us/ about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf.

[11] Blaise Gassend et al. “Silicon Physical Random Functions”. In: Proceedings of the 9th ACM conference on Computer and communications security. Washington, DC, USA, 2002, pages 148—160. doi: 10.1145/586110.586132.

[12] Larry Greenemeier. The Pentagon’s Seek-and-Destroy Mission for Counterfeit Electronics. 2017. url: https://www.scientificamerican.com/article/ the-pentagon-rsquo-s-seek-and-destroy-mission-for-counterfeit-electronics/.

[13] Jorge Guajardo et al. “FPGA Intrinsic PUFs and Their Use for IP Protection”. In: Cryptographic Hardware and Embedded Systems - CHES 2007. Ed. by Pascal Paillier and Ingrid Verbauwhede. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 63–80. isbn: 978-3-540-74735-2.

[14] Jorge Guajardo et al. “FPGA Intrinsic PUFs and Their Use for IP Protection”. In: Cryptographic Hardware and Embedded Systems - CHES 2007. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pages 63–80. isbn: 978-3-540-74735- 2.

[15] Ujjwal Guin et al. “Counterfeit Integrated Circuits: A Rising Threat in the Global Semiconductor Supply Chain”. In: Proceedings of the IEEE 102.8 (2014), pp. 1207–1228. doi: 10.1109/JPROC.2014.2332291.

[16] Basel Halak. Ageing of Integrated Circuits. 1st. Springer Nature Switzerland AG 2020, 2020. isbn: 978-3-030-23781-3.

[17] Charles Herder et al. “Physical Unclonable Functions and Applications: A Tutorial”. In: Proceedings of the IEEE 102.8 (2014), pages 1126–1141. issn: 1558-2256. doi: 10.1109/JPROC.2014.2320516.

[18] Benjamin Hill et al. “A split-foundry asynchronous FPGA”. In: Proceedings of the IEEE 2013 Custom Integrated Circuits Conference. 2013, pp. 1–4. doi: 10.1109/CICC.2013.6658536.

[19] Daniel E. Holcomb, Wayne P. Burleson, and Kevin Fu. “Power-Up SRAM State as an Identifying Fingerprint and Source of True Random Numbers”. In: IEEE Transactions on Computers 58.9 (2009), pages 1198–1210. issn: 2326- 3814. doi:

10.1109/TC.2008.212.

[20] “IEEE Standard for In-System Configuration of Programmable Devices”. In: IEEE Std 1532-2001 (Revision of IEEE Std 1532-2000) (2001), pages 1–158. issn: null.

[21] “IEEE Standard for Test Access Port and Boundary-Scan Architecture”. In: IEEE Std 1149.1-2013 (Revision of IEEE Std 1149.1-2001) (2013), pages 1– 444. issn: null. doi: 10.1109/IEEESTD.2013.6515989.

[22] Intrinsic-ID. https://www.intrinsic-id.com/. url: https://www.intrinsic-id.com/sram-puf/.

[23] Intrinsic-ID. Protecting a Device’s Root Secrets with SRAM PUF. url: https: //www.intrinsic- id.com/resources/videos/protecting- a- devicesroot-secrets-with-sram-puf-whiteboard-video/.

[24] N. G. Jacobson. “Streamlining programmable device and system test using IEEE Std 1532”. In: Proceedings International Test Conference 2000 (IEEE Cat. No.00CH37159). 2000, pages 847–853.

[25] Neil G. Jacobson, ed. The In-System Configuration Handbook: A Designer´s Guide to ISC. Norwell, Massachusetts, USA: Kluwer Academic Publishers, 2004. isbn: 140207655X.

[26] Marc de Jong and Anurag Srivastava. McKinsey on Semiconductors, Issue 7. 2019. url: https://www.mckinsey.com/industries/semiconductors/ our - insights / whats - next - for - semiconductor - profits - and - value - creation.

[27] A.B. Kahng et al. “Robust IP watermarking methodologies for physical design”.

In: Proceedings 1998 Design and Automation Conference. 35th DAC. (Cat. No.98CH36175). 1998, pp. 782–787. doi: 10.1145/277044.277241.

[28] Moon-Seok Kim et al. “Investigation of Physically Unclonable Functions Using Flash Memory for Integrated Circuit Authentication”. In: IEEE Transactions on Nanotechnology 14.2 (2015), pp. 384–389. doi: 10 . 1109 / TNANO . 2015 . 2397956.

[29] Paul Kocher, Joshua Jaffe, and Benjamin Jun. “Differential Power Analysis”. In: Advances in Cryptology — CRYPTO’ 99. Ed. by Michael Wiener. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 388–397. isbn: 978-3-540-48405-9.

[30] Paul Kocher et al. “Spectre Attacks: Exploiting Speculative Execution”. In: 40th IEEE Symposium on Security and Privacy (S&P’19). 2019.

[31] Farinaz Koushanfar. “Hardware Metering: A Survey”. In: ed. by Mohammad Tehranipoor and Cliff Wang. New York, NY, USA: Springer, 2012. isbn: 978- 1-4419-8079-3.

[32] Farinaz Koushanfar and Gang Qu. “Hardware Metering”. In: Proceedings of the 38th Design Automation Conference (IEEE Cat. No.01CH37232). 2001, pages 490–493. doi: 10.1145/378239.378568.

[33] Farinaz Koushanfar, Gang Qu, and Miodrag Potkonjak. “Intellectual Property Metering”. In: Information Hiding. Ed. by Ira S. Moskowitz. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 81–95. isbn: 978-3-540-45496-0.

[34] Sandeep S. Kumar et al. “Extended abstract: The butterfly PUF protecting IP on every FPGA”. In: 2008 IEEE International Workshop on Hardware-Oriented Security and Trust. 2008, pages 67–70. doi: 10.1109/HST.2008. 4559053.

[35] Lang Lin, Wayne Burleson, and Christof Paar. “MOLES: Malicious off-chip leakage enabled by side-channels”. In: 2009 IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers. 2009, pp. 117– 122.

[36] Moritz Lipp et al. “Meltdown: Reading Kernel Memory from User Space”. In: 27th USENIX Security Symposium (USENIX Security 18). 2018.

[37] K. Lofstrom, W.R. Daasch, and D. Taylor. “IC identification circuit using device mismatch”. In: 2000 IEEE International Solid-State Circuits Conference. Digest of Technical Papers (Cat. No.00CH37056). 2000, pp. 372–373. doi: 10.1109/ISSCC.2000.839821.

[38] Roel Maes and Vincent van der Leest. “Countering the effects of silicon aging on SRAM PUFs”. In: 2014 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST). 2014, pp. 148–153. doi: 10.1109/HST. 2014.6855586.

[39] David Manners. FBI arrests counterfeit chip traffickers. 2015. url: https:// www.electronicsweekly.com/news/business/fbi-arrests-counterfeit-chip-traffickers-2015-12/.

[40] Ravikanth Pappu et al. “Physical One-Way Functions”. In: Science 297.5589 (2002), pp. 2026–2030. issn: 0036-8075. doi: 10 . 1126 / science . 1074376. eprint: https://science.sciencemag.org/content/297/5589/2026.full. pdf. url: https://science.sciencemag.org/content/297/5589/2026.

[41] Kenneth Parker. The Boundary-Scan Handbook. 4th. Springer International Publishing Switzerland, 2016. isbn: 978-3319011738.

[42] Gang Qu and Lin Yuan. “Secure Hardware IPs by Digital Watermark”. In: ed. by Mohammad Tehranipoor and Cliff Wang. New York, NY, USA: Springer, 2012. isbn: 978-1-4419-8079-3.

[43] Jeyavijayan Rajendran, Ozgur Sinanoglu, and Ramesh Karri. “Is split manufacturing secure?” In: 2013 Design, Automation Test in Europe Conference Exhibition (DATE). 2013, pp. 1259–1264. doi: 10.7873/DATE.2013.261.

[44] Jeyavijayan Rajendran et al. “Security analysis of logic obfuscation”. In: DAC Design Automation Conference 2012. 2012, pp. 83–89. doi: 10.1145/2228360. 2228377.

[45] Ulrich R¨uhrmair, Srinivas Devadas, and Farinaz Koushanfar. “Security Based on Physical Unclonability and Disorder”. In: ed. by Mohammad Tehranipoor and Cliff Wang. New York, NY, USA: Springer, 2012. isbn: 978-1-4419-8079-3.

[46] Ulrich R¨uhrmair and Daniel E. Holcomb. “PUFs at a glance”. In: 2014 Design, Automation Test in Europe Conference Exhibition (DATE). 2014, pp. 1–6. doi: 10.7873/DATE.2014.360.

[47] Masoud Rostami, Farinaz Koushanfar, and Ramesh Karri. “A Primer on Hardware Security: Models, Methods, and Metrics”. In: Proceedings of the IEEE 102.8 (2014), pages 1283–1295. issn: 1558-2256. doi: 10.1109/JPROC.2014. 2335155.

[48] Jarrod A. Roy, Farinaz Koushanfar, and Igor L. Markov. “EPIC: Ending Piracy of Integrated Circuits”. In: 2008 Design, Automation and Test in Europe. 2008, pp. 1069–1074. doi: 10.1109/DATE.2008.4484823.

[49] PUF Security. pufsecurity.com. url: pufsecurity.com.

[50] COMMITTEE ON ARMED SERVICES UNITED STATES SENATE. INQUIRY INTO COUNTERFEIT ELECTRONIC PARTS IN THE DEPARTMENT OF DEFENSE SUPPLY CHAIN. 2012. url: https://www.armed-services.senate.gov/imo/media/doc/Counterfeit-Electronic-Parts. pdf

[51] SiidTech. www.siidtech.com. url: http://www.siidtech.com/howicidworks. html.

[52] Ying Su, Jeremy Holleman, and Brian P. Otis. “A Digital 1.6 pJ/bit Chip Identification Circuit Using Process Variations”. In: IEEE Journal of Solid-State Circuits 43.1 (2008), pages 69–77. issn: 1558-173X. doi: 10.1109/JSSC. 2007.910961.

[53] G. Edward Suh and Srinivas Devadas. “Physical unclonable functions for device authentication and secret key generation”. In: Proceedings of the 44th annual Design Automation Conference (2007), pages 9–14. doi: 10 . 1145 / 1278480.1278484.

[54] G. Edward Suh, Charles W. O’Donnell, and Srinivas Devadas. “Aegis: A Single-Chip Secure Processor”. In: IEEE Design Test of Computers 24.6 (2007), pp. 570–580. doi: 10.1109/MDT.2007.179.

[55] Mohammad Tehranipoor and Cliff Wang, eds. Introduction to Hardware Security and Trust. New York, NY, USA: Springer, 2012. isbn: 978-1-4419-8079-3.

[56] Laung-Terng (L.-T.) Wang, Xiaoqing Wen, and Khader S. Abdel-Hafez. "Chapter 2 - Design for Testability". In: VLSI Test Principles and Architectures. Ed. by Laung-Terng Wang, Cheng-Wen Wu, and Xiaoqing Wen. San Francisco: Morgan Kaufmann, 2006, pp. 37–103. isbn: 978-0-12-370597-6. doi: https: //doi.org/10.1016/B978- 012370597- 6/50006- 8. url: https://www. sciencedirect.com/science/article/pii/B9780123705976500068.

[57] James B. Wendt and Miodrag Potkonjak. "Hardware obfuscation using PUF-based logic". In: 2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). 2014, pages 270–271. doi: 10.1109/ICCAD.2014. 7001362.

[58] Neil H.E. Weste and David Money Harris. CMOS VLSI Design: A Circuits and Systems Perspective. 4th. Boston, Massachusetts, USA: Addison-Wesley Pearson Education, Inc., 2011. isbn: 978-0321547743.

[59] Meng-Yi Wu et al. "A PUF scheme using competing oxide rupture with bit error rate approaching zero". In: 2018 IEEE International Solid - State Circuits Conference - (ISSCC). 2018, pp. 130–132. doi: 10.1109/ISSCC.2018. 8310218.

[60] Xilinx. "7 Series FPGAs Configuration User Guide (UG 470)". In: (2018).

[61] Xilinx. "XA Artix-7 FPGAs Data Sheet: Overview (DS197)". In: (2017).

[62] Xilinx. "Xilinx Zynq-7000 SoC Data Sheet: Overview (DS190)". In: (2018).