8-1-2017

Design and Architecture of Hardware-based Random Function Security Primitives

Fatemeh Tehranipoor
University of Connecticut - Storrs, fatemeh.tehranipoor@uconn.edu

Design and Architecture of Hardware-based Random Function Security Primitives

Fatemeh Tehranipoor, Ph.D.

University of Connecticut, 2017

In recent years, security has grown into a critical issue in modern information systems. Electronic hardware security, in particular, has emerged as one of the most serious challenges due to electronic devices penetrating every aspect of our society. Furthermore, due to the trend in globalization, system integrators have had to deal with integrated circuit (IC)/intellectual property (IP) counterfeiting more than ever. These counterfeit hardware issues counterfeit hardware that have driven the need for more secure chip authentication, since traditional ID or key storage have been demonstrated to be vulnerable to various kinds of attacks. In addition, due to the need for highly secure electronic information systems, almost every important and valuable document or piece of data is stored/transferred in

some type of encrypted form to prevent attackers from compromising privacy or stealing information for nefarious uses. High entropy random numbers from physical sources are a critical component in authentication and encryption processes within secure systems. Secure encryption is dependent on sources of truly random numbers for generating keys, and there is a need for an on chip random number generator to achieve adequate security. Furthermore, the Internet of Things (IoT) adopts a large number of these hardware based security and prevention solutions in order to securely exchange data in resource-efficient manner. Note that due to the nature of IoT systems, these networked devices are particularly vulnerable to attacks that involve physical manipulations. In this work, we have developed several methodologies of hardware based random functions in order to address the issues and enhance the security and trust of ICs. The methodologies proposed in this thesis include: a novel DRAM-based intrinsic Physical Unclonable Function (PUF) for system-level security and authentication along with analysis of the impact of various environmental conditions, particularly silicon aging; a DRAM remanence based True Random Number Generation (TRNG) to produce random sequences with a very low-cost overhead; a DRAM TRNG model using its startup value behavior for creating random bit streams; an efficient power-supply noise based TRNG model for generating an infinite number of random bits which has been evaluated as a cost effective technique; architectures and hardware security

solutions for the Internet of Things (IoT) environment. Since IoT devices are heavily resource-constrained, our proposed designs can alleviate the concerns and issues of establishing trustworthy and secure systems in an efficient and low-cost manner.

Design and Architecture of Hardware-based Random Function Security Primitives

Fatemeh Tehranipoor

B.S., University of Mazanadaran, Mazanadaran, Iran, 2011

M.S., Shahid Beheshti University, Tehran, Iran, 2013

A Dissertation

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

at the

University of Connecticut

2017

APPROVAL PAGE

Doctor of Philosophy Dissertation

Design and Architecture of Hardware-based Random Function Security Primitives

Presented by

Fatemeh Tehranipoor

Major Advisor

John A. Chandy

Associate Advisor

Lei Wang

Associate Advisor

Faquir Jain

University of Connecticut

2017

This work is humbly dedicated to my valuable treasures in life:

To my husband and my parents

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisor Dr. John Chandy for the continuous support of my Ph.D study and related research for his patience, motivation, and immense knowledge. I am very grateful for his support in overcoming numerous obstacles I have been facing through research. His guidance helped me in all the time of research and writing this thesis. He also was a great role model both in and out side of academia. I could not have imagined having a better advisor and mentor for my Ph.D study.

Besides my advisor, I would to thank the rest of my thesis committee: Dr. Lei Wang , Dr. Faquir Jain, Dr. Rajeev Bansal, and Dr. Helena Silva for their insightful comments and encouragement.

Furthermore, I thank my labmates for the stimulating discussions, for the times that we were working together before impossible deadlines when we did not even have a choice, even if that meant switching between multiple projects in one day, and especially for all the fun we have had in the last two years.

Last but not the least, I would like to thank my dearest family from the bottom of my heart: my wonderful husband and my great parents for their love, support, and encouragement. It is my greatest honor to dedicate this work to them.

TABLE OF CONTENTS

1. Introduction 1

1.1 Security Vulnerabilities 3

1.2 Hardware Security Primitives 4

1.2.1 Physical Unclonable Functions 5

1.2.2 Random Number Generations 8

1.2.3 Security Primitives Potential Application Areas 10

1.3 Problem Statement 11

1.4 Related Works and Limitations 12

1.5 Contributions 16

1.5.1 PUF Design 16

1.5.2 RNG Design 16

1.5.3 Primitive Designs in IoT Environment 17

1.6 Thesis Outline 17

2. DRAM-based Intrinsic Physically Unclonable Functions 19

2.1 Introduction 19

2.2 Security Level Security 20

2.3 DRAM PUF Description and Properties 20

2.3.1 DRAM PUF Advantages 21

2.3.2 Potential DRAM PUF Implementations 22
2.3.3 Startup Value Based DRAM PUF 23
2.3.4 Uniformity of Memory 24
2.4 Experimental Setup 26
2.5 DRAM PUF Evaluation under Different Environmental Operating Conditions 28
2.5.1 Stability of DRAM under Temperature Variation 29
2.5.2 Stability of DRAM PUF under Voltage Variation 31
2.5.3 Stability of DRAM PUF due to Aging 32
2.6 Bit Selection Algorithm for Generating PUF IDs 33
2.7 Analysis of Experimental Results and Their Validation 35
2.7.1 Multi-device Evaluation on DRAMs 37
2.7.2 Reliability 37
2.7.3 Our Selection Algorithm vs Baseline Algorithm 41
2.7.4 Uniqueness 41
2.7.5 Randomness 43
2.7.6 Case Studies 45
2.8 DRAM PUFs Reliability Analysis due to Device Accelerated Aging . 47
2.8.1 PUFs under Accelerated Aging 47
2.8.2 Experimental Platforms 51

2.8.3 DRAM PUFs Aging Effects Study and Procedure 51
2.8.4 Reliability of DRAM PUFs under Aging Condition 53
2.9 DRAM PUF as a System Security Solution 57
2.10 Conclusion 60

3. DRAM-based Random Number Generation Design 62
3.1 Robust Hardware True Random Number Generations using DRAM Remanence Effects 62
3.1.1 Introduction 62
3.1.2 Data Remanence 63
3.1.3 DRAM Remanence-based TRNG 64
3.1.4 Potential Attacks 71
3.1.5 Data Analysis and Evaluation 71
3.2 DRAM-based Random Number Generation using its Startup Value Behavior 77
3.2.1 DRAM Startup Value Behavior 77
3.2.2 Experimental Setup 77
3.2.3 Data Analysis and Results 78
3.3 Potential Attacks 88
3.4 Conclusion 88

4. A Study of Power Supply Variation as a Source of Random Noise 90

4.1 Introduction 90

4.2 Challenges, Motivations, and Objectives 91

4.3 Power Supply Noise under Study 93

4.4 Dynamic Variation in Power Supplies vs Voltage Regulators 93

4.5 Preliminary Power Supply-based TRNG Circuit Model 101

4.5.1 Simulation Results 102

4.5.2 Implementation Results 103

4.5.3 Experimental Setup 106

4.6 Advanced Power Supply-based TRNG Model using Tuning System . 106

4.6.1 Tuning System 106

4.6.2 Dynamic Voltage Feedback Tuning (DVFT) Design 108

4.6.3 Data Analysis and Results 117

4.6.4 Attacks and Defenses 123

4.7 Implementation of Power Supply-noise based TRNG in Linux Operating System 127

4.7.1 Functionality of the Proposed TRNG Model in Linux 128

4.7.2 Experimental Setup 128

4.7.3 Randomness Test 129

4.8 Conclusion 129

5. Hardware Security Architecture for the Internet of Things (IoT) 132

5.1 Introduction 132

5.2 IoT Adoption and Expansion 133

5.3 Security and Privacy Challenges in IoT Systems 134

5.4 From Hardware Security Primitives to the Internet of Things 136

5.5 Case Study 1: Our Proposed Architecture for Authentication in IoT Healthcare Applications 138

5.5.1 Infrastructure Security In Healthcare 139

5.5.2 Motivation and Challenges 140

5.5.3 Proposed Security Approach for Authentication in IoT Healthcare . 141

5.5.4 Examining Security of our Architecture Model 150

5.5.5 Case Study 2: Examination of using Human Characteristics (Biometrics) for Authentication in a Smart DoorLock IoT Device 151

5.6 Conclusion 156

6. Conclusions 158

6.1 DRAM-based Physical Unclonable Functions 159

6.2 DRAM-based Random Number Generations 160

6.3 Power Supply Noise-based True Random Number Generations 161

6.4 Hardware Security Architectures for the Internet of Things 162

6.5 Future Work 163

6.6 Summary of Conclusions 164

Bibliography 165

A. Related Publications 177

LIST OF FIGURES

1.1 Physical Unclonable Functions Secure Authentication Mechanism . . 8

1.2 General schematic of a TRNG model. 10

1.3 PUFs Taxonomy based on Origin of Stimulus. 13

2.1 Memory structure of a One-Transistor DRAM array. 25

2.2 Timing diagram of a DRAM read operation of an uncharged cells biased to Vdd (a) or Vss (b) due to process variations. 26

2.3 Uniformity of DRAMs across 10 measurements for each of 8 DRAMs (average percentages of 1s values) 27

2.4 Experimental setup with Xilinx Spartan-6 FPGA (right side) under the test using the ThermoStream system for high and low temperature variations. 30

2.5 Schematic of grids (rows and columns) on DRAM cells. 35

2.6 Muti-device evaluation; (a1) & (a2), (b1) & (b2), and (c1) & (c2) show the stability (percentages of '1's and '0's) across different set of measurements for DRAM1, DRAM2, DRAM3 respectively. . . 38

2.7 Distribution of Intra-die Hamming Distance (HD) among 48 (3×16) DRAM-based PUFs under different operating conditions. 39

2.8 Distribution of Inter-die Hamming Distance (HD) of 3 DRAMs among Different the extracted IDs. 44

2.9 Percentage of bit flips across multiple measurements (a) under accelerated aging condition (Bit Selection Algorithm), (b) under random condition (Random Bit Selection) 54

2.10 Percentages of HDs of DRAMs among different IDs under Aging condition in (a) Sep. 2014, (b)Feb. 2015, (c) Mar. 2015, (d) Apr. 2015, (e) Jul. 2015 (f) Aug. 2015, (g) Jan. 2016, and (h) Feb. 2016. 58

3.1 DRAM cells array with a typical single MOSFET transistor and a storage capacitor. 66

3.2 Real data snippets that illustrate the DDR2 SDRAM operations: a) Write. b) Delay(1). c) Read(1). d) Read(m). 67

3.3 Our DRAM Remanence based TRNG Model using MATLAB Curve Fitting Tool (CFTool). 69

3.4 The histogram of the XORed data between Fluctuation and Startup regions. 73

3.5 Scatter plot of percentage of '1's of XOR'ed data that are taken from Fluctuation and Startup regions. 74

3.6 The histogram of the Uniqueness of the data at Fluctuation Region. . . . 75

3.7 Small section of the bitmap from DRAM for trial 3. 78

3.8 Small section of the bitmap from DRAM for trial 6. 79

4.1 Histograms of a portion of the data (10000 samples) directly from six different power sources (Bench power supply, USB, Computer power source, DC power supply). 94

4.2 Input voltage plot of 1-second (240 samples) of the data from 6 different power supplies. 95

4.3 Using voltage regulator (LT3501EFE#PBF) for plotting: (a) Histogram of data (10000) samples, (b) Input voltage of 1-second (240 samples) of the data. 96

4.4 A schematic of our TRNG circuit design. 99

4.5 Plots of output voltage vs the number of samples while using (a) 1-inv, (b) 2-inv, (c)3-inv, and (d) 4-inv in chain using Monte Carlo simulation. 100

4.6 Percentages of 1's at different input voltage level (0-5 volts) after using different number of inverters in inverter chain (1-5 inverters) from 6 different power supplies. 105

4.7 DVFT circuit model for the proposed TRNG. 110

4.8 Timing diagram of different points (p0, p1, p2, p3) of our model. . . 113

4.9 Voltage sweeping using Arduino as a variable load to the input. . . . 121

5.1 Gartner 2012 Hype Cycle of emerging technologies. 134

5.2 A platform of IoT-based ubiquitous healthcare solutions using secure hardware elements for Authentication. 142

5.3 Security Approach for Low-Cost Verification of Devices and Patients in the healthcare domain. 145

5.4 Schematic for key generation procedure. 150

5.5 Illustration of additional authentication methods to existing smart lock model (Kwikset Kevo). 155

LIST OF TABLES

2.1 DRAMs Stability across Different Nominal Conditions 29

2.2 Stability under Temperature Variations compared to Nominal Conditions . 30

2.3 Stability under Voltage Variations compared to Nominal Conditions . . . 31

2.4 Stability under Aging Condition compared to Nominal Conditions . . 33

2.5 Percentage of bit flips across multiple measurements (10-Sets) under Baseline (Base), Neighbor Selection (NS), Environmental Screening (ES), and combined (Algo) approaches. 40

2.6 Percentage of bit flips across multiple measurements under normal operating conditions. 40

2.7 Quality Evaluation of IDs from DRAM PUFs. 47

2.8 Stability of DRAMs due to Aging compare to Nominal Conditions . . 53

3.1 NIST Statistical Tests Results at Room Temperature Condition. . . . 76

3.2 DRAM trials and corresponding NIST tests 82

3.3 DRAM trials with Von Neumann corrector 84

3.4 DRAM trials XOR with Von Neumann corrector 86

4.1 Power sources under evaluation 98

4.2 NIST Statistical Tests Suite average p-value results of our proposed TRNGs using different types of power supplies. 107
4.3 NIST Statistical Tests Suite average p-value results of our proposed TRNG model using feedback loop (DVFT) implemented with different power supplies. 122
4.4 NIST Statistical Tests Suite Results 130

Chapter 1

Introduction

As electronic devices become ubiquitous and more interconnected, people must depend on Integrated Circuits (ICs) for the security of sensitive information. Providing this security relies on well-established primitives for key generation, data confidentiality and integrity, authentication, identification bit commitment, etc. Therefore, it is paramount for ICs to be able to perform operations and critical tasks in a low-cost yet highly secure way. Unfortunately, the conventional approaches (e.g. digital signatures, encryption) suffer from various shortcomings; they are very slow, expensive, and increasingly vulnerable to physical and side channel attacks. Hardware-based security primitives such as physically unclonable functions (PUFs) and true random number generators (TRNGs) can overcome these limitations and provide random functions in order to establish security and trustworthiness in critical application and systems. PUFs can derive secrets from the complex physical characteristics of ICs rather than storing the secrets in digital memories. PUFs can significantly increase physical security by generating

volatile secrets (keys) that only exist in a digital form when an IC is powered on and operating. Furthermore, a TRNG is an important security primitive used in a variety of applications including cryptographic algorithms, statistics, communication systems, simulations, etc. It is critical that a TRNG be able to produce outputs consisting of fully unpredictable and unbiased bits in a cost-effective manner. In general, these hardware security primitives should provide low-cost and efficient trustworthiness of the physical hardware platforms. One should note that while these primitives can provide advantages to ICs, there are properties and details of the design that need to be considered (e.g. power usage, overhead, heat).

Since the Internet of Things (IoT) is a rapidly emerging paradigm, the most demanding requirement for their widespread realization is security. Applying low-cost security solutions to a large scale of IoT [88] and even Cyber-Physical Systems (CPSs) [18] is possible using hardware-based security primitives such as PUFs and TRNGs. Providing a secure framework and platform for IoT systems can protect them against malicious attacks. One of the most challenging concerns for developing secure IoT devices is the resource constrained nature of these embedded systems. Security traditionally requires a great deal of resources in order to perform the computations necessary for encryption, certificate verification, third-party authentication, etc. By implementing the previously discussed

hardware security primitives, developers can easily overcome the issues of resource constrained IoT device trustworthiness and verifiability in a low-cost and efficient way.

1.1 Security Vulnerabilities

In recent years, security has grown into a critical issue in modern information systems. Electronic hardware security, in particular, has emerged as one of the most serious challenges due to electronic devices penetrating every aspect of our society. Due to globalization trends, intellectual property (IP) vendors and system integrators have to deal with various counterfeiting issues more than ever and this surge in counterfeit hardware has driven the need for more secure chip authentication. Among the sources of counterfeit chips are reintroduced discarded chips into the supply chain and fabrication of cheap copies that pass as authentic without significant scrutiny. Since the IP owner cannot be present during the fabrication process, this makes Integrated Circuit (IC) designs increasingly vulnerable to malicious modifications. In today's technology-based day to day existence, there is a need to maintain the privacy of users and their data. These efforts are necessary to protect oneself against an unending tide of malicious activity and eavesdroppers. To complete this task one requires large random bit strings of information that can be applied for encoding, or decoding, sensitive information. The randomness

of the bit strings used allows a user to counter the ability of an attacker to predict the result of the data manipulation. Otherwise a nefarious individual would be able to decrypt secure communications with minimal effort due to function-based (e.g. time domain) knowledge. As a rule of thumb, the algorithms used to produce these random numbers must have fast speed, require low power to operate, provide high reliability, and have high entropy.

1.2 Hardware Security Primitives

Traditional mathematical cryptography methods relies on the existence of one or more pieces of secret information (keys) to perform a secret exchange of information. The mathematical properties of the cryptographic algorithms ensure the infeasibility of mathematical attacks on the ciphers. However, these algorithms depend on the secure storage of these secret keys. With the ubiquitous deployment of cryptographic hardware and software secret storage of keys is not guaranteed because of the ability to easily attack non-volatile storage such as flash. Thus, alternative cryptographic protocols based on hardware security primitives that do not require an explicit secret key storage.

1.2.1 Physical Unclonable Functions

As a means to uniquely identify chips, researchers have proposed using the random process variations that naturally occur during the manufacturing process. These effects include process variations such as the size of transistors, capacitors, resistors and other components. These are unavoidable for the most part, and must be accounted during the design and layout process. However, these random process variations can be used to our advantage if we use them to generate unique intrinsic identifiers. This is the idea behind Physically Unclonable Functions (PUFs), which was first proposed by Gassend et al. in 2002 [29]. Gassend and Pappu in 2001 developed the first silicon PUFs through the use of intrinsic process variation in deep submicron integrated circuits. They used the intrinsic process variability of silicon devices during manufacturing to produce unique, random and unclonable digital responses and called it a physically random function. They have since been called physical unclonable function to emphasize the fact that they are not repeatable. Generally speaking, PUFs should present unpredictable, robust and unclonable characteristics. PUFs are circuits that have come into prominence in the past decade and hold much promise as a hardware security primitive.

A PUF can provide a hardware specific unique signature or a "fingerprint" for an integrated circuit that can be leveraged for various security applications including authentication and secure access. One strong characteristic of a PUF is

that it can not be reverse engineered easily. There are two fundamental requirements for building a PUF: *random* and *uncontrollable* variations. The variations must be random, thereby drastically reducing the probability that a unique signature will be repeated. Also, the variations must be uncontrollable such that an adversary can not clone the devices. A PUFs inputs and outputs map a specific set of challenges to a set of corresponding responses which are called Challenge-Response Pairs (CRPs) [61]. In other words, a PUF is a multiple-input (challenges) multiple-output (responses) function that has hard-to-predict dependency between the outputs and its inputs. The functional relationship between challenge and response looks like that of a random function. Because the PUF is derived from random process variation, it is very difficult, if not impossible, to predict the responses from a particular challenge or construct a function to do so in hardware or in software.

In general, a good PUF should have several parameters and characteristics, in particular Uniqueness, Reliability, and Randomness. Uniqueness of a PUF represents the ability of that PUF to uniquely generate responses. In other words, PUF uniqueness means that different PUFs generate different responses for the same challenge. Reliability of a PUF demonstrates that a given PUF can regenerate the same response for a particular challenge consistently. Finally, the randomness of a PUF indicates how random the response bits are. Ideally, the

response should follow a uniform distribution, whereby the proportion of 0s and 1s in the response bits should be equal. Figure 1.1 shows a PUF-based secure authentication technique. For simple IC authentication, PUF challenges (input) - response (output) pairs are collected from each chip, and stored in a secure database. In order to authenticate any given IC, one of the challenges is presented to the IC, the PUF embedded in the IC generates a response. If the produced response matches the one that is stored in the database, that means the IC is authentic. Note that in order to prevent man-in-the-middle (MiTM) attacks, each CRP (challenge-response pair) is used only once. An application for PUFs is that they can be used to provide secure, safe and low-cost authentication by generating unique secret keys/IDs. The ability of PUF devices to provide bit strings unique to each component can be leveraged as an authentication mechanism to detect tampering and attacks [16].

The advantage of using PUFs compared to traditional solutions is that they are: highly secure and inexpensive, because they provide volatile secrets which do not need to be stored anywhere and do not require any special fabrication technology and manufacturing process. Since PUF application is typically characterized by having two phases, enrollment and regeneration, it has the potential to be used for authentication purpose easily.

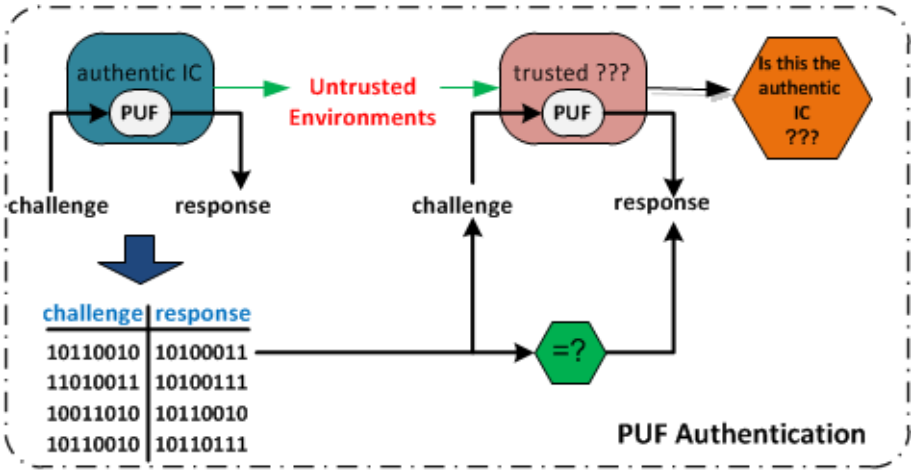


Fig. 1.1: Physical Unclonable Functions Secure Authentication Mechanism

1.2.2 Random Number Generations

Cryptography and security applications make extensive use of random numbers and random bits. Random numbers are useful for a variety of purposes, such as generating data encryption keys, simulating and modeling complex phenomena, selecting random samples from larger data sets, and even for gambling. Random number generators (RNGs) are classically divided into two different types: Pseudo random number generators (PRNGs) and True random number generators (TRNGs). PRNGs are deterministic in nature, but are traditionally adequate for most applications. These type of random number generators usually require a seed (i.e. number to initialize the internal state of the generator) and the seed should be periodically changed to keep the system secure. The number sequence produced by PRNGs is random within a specific time period; meaning the method of ran-

dom number generation does not provide truly random behavior. TRNGs, on the other hand, derive their randomness from a physical entropy source and provide inherently nondeterministic behavior. They are unpredictable, and are random in the entire time domain. Since TRNGs are capable of producing uncorrelated and irreproducible procedures they act as a critical component within cryptographic systems and applications. For security-centric applications the high entropy numbers from physical sources are a critical component in authentication and data encryption processes, where they are used to generate random cryptographic keys that are used to transmit data securely. Designing TRNGs around new forms of noise, one must account for certain features. Ideal TRNGs should display three essential characteristics: efficiency, non-determinism and non-periodicity. The dynamically natured variations that are induced by power supply noise exhibit the necessary characteristics. A general schematic of a TRNG model is shown in 1.2. One can see that the model has a physical random source. There is also a data sampling module that can produce raw bits. If a user requires post-processing (e.g. Von Neumann [44]) then the model can produce processed data, at the cost of additional overhead to the model's operation.

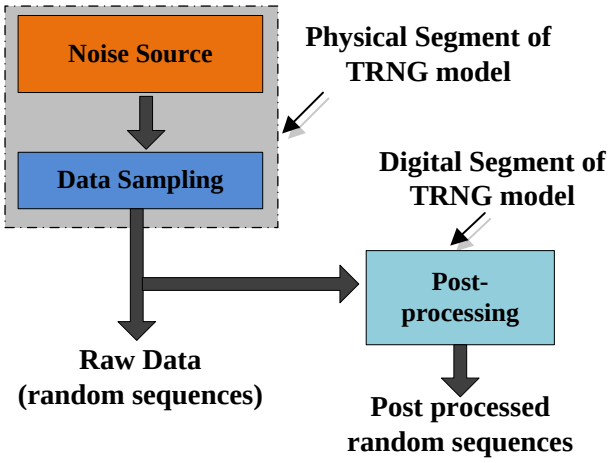


Fig. 1.2: General schematic of a TRNG model.

1.2.3 Security Primitives Potential Application Areas

Establishing security across physical hardware platforms can be a resource intensive and costly aspect when protecting the storage and exchange of data between components. As discussed in previous sections, one can use these security primitives for the purpose of securing physical objects to protect sensitive data and functionality of a device from a malicious actor. PUFs allow for unique identity to be attached to specific components or systems. This functionality provides the ability to authenticate hardware by producing individual signatures (IDs) for an implemented IC. It also can generate keys that can be used for encryption/decryption algorithms. This is possible since the randomness of the produced IDs/keys are enough to be used for this purpose. As TRNGs are another security primitive that are uniquely tailored to produce high random sequences of bits for a

multitude of applications. These applications include: gaming, product labeling, simulations, cryptography and etc. The reason that these security primitives are widely adopted is they exploit the physical properties of a hardware system to produce random output. Due to this popularity, these solutions have the potential to be used in securing IoT environments as long as their design is focused on being low-cost and resource efficient.

1.3 Problem Statement

Reliability and uniqueness are always important issues that hinders PUFs practical applications. The stability of PUFs under various operating conditions has been a serious concern facing different kinds of PUFs. Another largest problems facing the continued use of hardware-based random functions (PUFs and RNGs) is that as technology continues to grow, there is a high demand for low-cost resource efficient solutions. In order to overcome this potential future limitation, we proposed novel designs and architectures for the improvement of hardware-based random functions that meet the desired requirements of new embedded system security needs. Our solutions to this problem, which we will talk about in the following chapters, can be used to provide physical objects security in a variety of different applications such as the Internet of Things.

1.4 Related Works and Limitations

(1) PUFs: Since PUFs have gained considerable attention in the past few years, it has yielded several proposed approaches for the realization of these functions. Figure 1.3 shows a comprehensive taxonomy of existing PUF types and categorization. The two largest categories of PUFs are intrinsic and extrinsic. Extrinsic PUFs rely on some external stimuli in order to generate CRPs, for example light for an optical PUF. On the other hand, intrinsic PUFs are ones that depend on the natural internal manufacturing property (process variation) of the device. Examples of intrinsic PUFs include delay based PUFs (arbiter PUF, RO PUF) which can depend on digital race condition or frequency oscillation (delay), respectively [40]. So far, various kinds of PUFs have been proposed for key generation/ID, such as RO-PUF [114], [60], [116], [111], [108], Arbiter PUF [94], Butterfly PUF [69], Clock PUF [113], Controlled PUF [20], Rowhammer PUF [90], Memristor PUF [68], etc. Many methods have already been proposed for identification and authentication of ICs such as in [94], [73]. Of particular interest are memory-based PUFs, which are attractive because most electronic systems have some type of memory included. The limitations of PUFs are that most previous PUF designs suffer from adding extra circuit to the existing hardware. However, memory-based PUF designs do not need any extra circuitry to achieve reliable PUF IDs/keys. Memory-based PUFs are usually based on the measurement of startup values of memory cells.

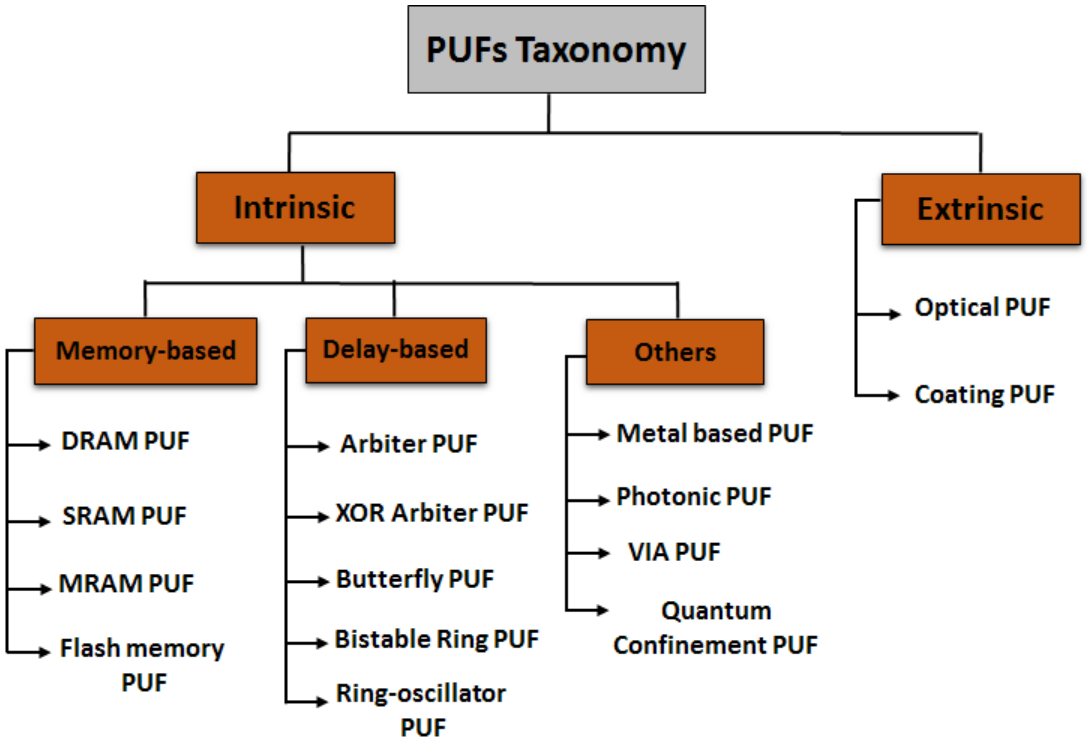


Fig. 1.3: PUFs Taxonomy based on Origin of Stimulus.

Flash memory is a Non-Volatile Memory (NVM) that has been proposed as a memory-based PUF in [102] and [77]. SRAM PUFs are one existing memory based PUF which has been presented in [106], [76], [91], [41], [42]. A SRAM PUF can generate a device-individual fingerprint using the startup behavior of its cells. SRAM PUFs exploit the inherent threshold variation of the cross-coupled SRAM cells. DRAM PUF is a newly presented work by Hashemian et al. [37], Keller et al. [52], Sutar et al. [95], Xiong et al. [107], Liu et al. [57]. Researchers have also proposed potential PUFs using future memory technologies such as memristors [86], [53], spintronic memories [43], MRAM-based PUFs [98], [21], etc.

(2) RNGs: Generally, PRNGs fit most of the application needs but there are demanding situations where PRNGs are substituted by TRNGs and those are the applications where it is important that the numbers be really unpredictable, such as for data encryption, and generating cryptographic keys. The reason being the PRNGs lack of strong randomness properties [15], which can be seen via various statistical tests. Hence, the output sequences from TRNGs should have good statistical properties verified with the use of statistical tests, e.g. from NIST 800-22 statistical test suite [87]. However, the generation of true random bits is problematic in many practical applications of cryptography. Currently, several techniques are used for implementing TRNGs. In [12], Bruynincks et al. analyzed the security requirements of TRNGs, demonstrated the real-life attacks performed on various types of TRNGs and proposed solutions for generating safe cryptography random bits using TRNGs from untrusted vendors. In [104], the authors exploited random behavior from nearly-metastable operation of a group of FPGAs. An oscillator-based TRNG has been proposed in [3] that can automatically adjusts the generated unbiased random numbers produced by process variation and dynamic temperature. Mudit et al. proposed a TRNG design based around sense amplifier circuits that are balanced in the metastable region using hot carrier injection in [9]. Recently, new TRNG models such as Technology Independent (TI) TRNG, TRNG using hot-carrier injection balanced metastable sense ampli-

fiers, Portable TRNG for personal encryption application based on smart phone cameras, and Highly Efficient TRNG in FPGA Devices Using Phase-Locked Loops are investigated in [81], [9], [115], and [22], respectively.

Various types of TRNGs have been proposed using different noise and random sources such as ring oscillator (RO) TRNGs [24] [56] [82], memory based TRNGs such as flash memory based [102] [19], SRAM-based TRNG [79], block memory based TRNGs [33], metastability-based TRNGs [105], emerging device noise based TRNGs [17] [38], etc. Some hardware-based TRNGs use high gain amplifiers to enhance a noise source such that it can be discriminated by a comparator [75], [2]. Burleson et al. did some analysis of on-chip true random number generators based on power supply variation [13].

The issue when examining the challenges of previous TRNG work is that there is a clear line of acceptance between a working TRNG and a non-working TRNG device. There is, however, a range of success for TRNG models that meet the required standard. Some TRNGs have better results in terms of randomness while others do not. The main challenge in TRNG research include low-cost design that are robust against manipulation of environments (voltage and temperature) and silicon aging. Our proposed TRNG models (DRAM remanence-based TRNG (chapter 3) and Power supply-noise based TRNG (chapter 4)) have the advantage of being very cost-effective and lightweight in terms of overhead. We have

proposed a remanence based TRNG that has the advantage of being simplistic, in that, our model does not need any extra hardware or overhead when compared to existing solutions for TRNGs. Furthermore, our power supply noise-based has the advantage of producing infinite number of random sequences.

1.5 Contributions

This thesis is devoted to the design and architecture of various hardware-based random number function security primitives that meets the requirements and needs for low-cost solutions in today's electronic systems. Our work will also deal with the architectural requirements for designing resource efficient embedded system platforms. Specifically, our contributions are as follows:

1.5.1 PUF Design

We have developed new DRAM-based Physical Unclonable Functions using its startup value behavior. Furthermore, we evaluated silicon aging effects on DRAM PUFs reliability. Overall, developing PUF design for optimizing the use of existing hardware resources for the purpose of creating random output was our goal.

1.5.2 RNG Design

We have developed random number generators particularly TRNGs based on existing memories (DRAM-based TRNG) and power supply-noise based true ran-

dom number generator. Our goal while working on RNG-related projects was to develop and introduce TRNG designs for the exploitation of the existing hardware platform characteristics to generate highly random bit streams.

1.5.3 Primitive Designs in IoT Environment

After producing improved designs of our hardware-based random functions and incredible results, we found that our developed solutions minimized additional cost overhead and made optimal use of system resources. We noticed that the majority of hardware-based random function solutions for distributed embedded systems environment (i.e. IoT) were not designed to be low-cost and resource efficient. We performed case studies to determine the effectiveness of our hardware security primitive solutions in IoT healthcare domain and smart device authentication (Smart DoorLock).

1.6 Thesis Outline

The outline of this thesis is as follows:

- Introduction
- DRAM-based Intrinsic Physically Unclonable Functions
- DRAM-based Random Number Generation Design

- Power Supply Noise-based True Random Number Generation
- Hardware Security Architectures for the Internet of Things (IoT)
- Conclusions

Chapter 2

DRAM-based Intrinsic Physically Unclonable Functions

2.1 Introduction

In this chapter, we introduce an intrinsic PUF based on dynamic random access memories (DRAM). DRAM PUFs can be used in low cost identification applications and also have several advantages over other PUFs such as large input patterns. The DRAM PUF relies on the fact that the capacitor in the DRAM initializes to random values at startup. We demonstrate real DRAM PUFs and describe an experimental setup to test different operating conditions on three DRAMs to achieve the highest reliable results. Furthermore, we select the most stable bits use as chip ID using our enrollment algorithm. We also evaluate silicon aging effects on DRAM PUFs in details. In other words, we explore the possibility of intrinsic PUFs within Commercial Off-The-Shelf (COTS) DRAM ICs. We describe how to use the signatures to prevent modifications and uniquely identify and/or authenticate electronic devices.

2.2 Security Level Security

Most electronic systems are not designed with security in mind, as a result, there are always threats from attackers to alter these systems and leak secret information from them. Even if the systems are securely designed, there is no assurance that the delivered system is authentic. System level security mechanisms can use a subsystem on the board to prevent any altering or modification in system functionality and stop or reset system if any anomalous behavior has been detected. While it is difficult to authenticate the trustworthiness of any particular IC on a system board, a unique identifier, such as a PUF embedded in an IC can be used and gives the IC a unique identity. However, with COTS parts, a PUF or chip ID may not be available, so mechanisms for intrinsic PUF identification are needed. In this chapter, we present an intrinsic DRAM PUF that can be used to authenticate electronic systems on which DRAMs are present.

2.3 DRAM PUF Description and Properties

PUFs intrinsic to DRAM ICs have not been explored extensively. Our primary contribution is the identification of a DRAM PUF based on start up values. We examine the effect of various operating conditions such as temperature variation, voltage variation, and aging which may influence the behavior of the DRAM PUF. Finally, we propose a selection mechanism to isolate highly stable bits within the

large set of available bits in a DRAM.

2.3.1 DRAM PUF Advantages

DRAMs have some unique advantages that motivated us to explore it further:

1. **Large input pattern:** Because of the large number of available bits in a typical DRAM, one can generate a large set of input challenges and correspondingly large output responses. This characteristic of DRAM PUF is very valuable which can make it to be distinct among all kinds of intrinsic PUFs.

2. **Cost-effective:** Since many computer systems have some form of DRAM on board, DRAMs can be used as an effective system-level PUF as well. It is also much cheaper than SRAM. Thus, DRAM PUFs could be a source of random but reliable data for generating board identifications (chip ID). The advantage of the DRAM PUF is based on the fact that the stand-alone DRAM already present in a System on a Chip (SoC) can be used for generating device specific signatures without requiring any additional circuitry or hardware [37]. PUFs intrinsic to DRAM ICs have not been explored extensively. Ours is one of the first works in which a DRAM has been used as a system level security Physical Unclonable Function.

2.3.2 Potential DRAM PUF Implementations

DRAM memory cells are comprised of a paired transistor and capacitor. While ideally every DRAM cell should be identical, manufacturing imperfections cause slight physical variations in each cell. Moreover, every DRAM cell has its own physical trait. Therefore, the leakage effects on the storage nodes will vary as well. These physical variation characteristics can be potentially used to develop PUFs. The only previous work on DRAM PUF has been based on altering or disabling the refresh cycle [52]. Modern DRAM chips have a built in self-refresh module, as they not only require a power supply to retain data, but must also be periodically refreshed to prevent their data contents from fading away from the capacitors in their integrated circuits. The essential approach with refresh-based DRAM PUFs is to initialize all cells to 1 and then after some time, with refresh turned off, some of the cells will leak to 0. The randomness of which cells leak to 0 provide the opportunity for a PUF. The difficulty with using these refresh or retention based methods for a PUF is that it may take several minutes to hours for sufficient cells to flip to 0. Another potential approach is to use the remanence property of DRAMs. Contrary to popular belief, DRAMs can hold their values for surprisingly long intervals without power. DRAM cells retain their contents for a few seconds to minutes at room temperature. In fact, it has been demonstrated that sensitive information can be extracted from volatile memories due to data

remanence effects [31], [35]. Based on our examination on DRAMs, remanence approach is not feasible for constructing PUFs, however remanence effect can be used for creating True Random Number Generators (TRNGs).

2.3.3 Startup Value Based DRAM PUF

In our observation of DRAM refresh and remanence properties, however, we noticed that certain DRAMs actually exhibit behavior similar to SRAMs, i.e. they have seemingly random startup values. In other words, the cells do not initialize to '0' as would be expected. Thus, as with SRAMs, these startup values provide a potential for creating a PUF. The reason for this random startup behavior can be explained by the interaction of precharge, row decoder, and column select lines when the device is powered up. Figure 2.1 shows the structure of a typical DRAM array. Bits are stored either by charging the storage capacitor to V_{DD} or discharging it to ground. The timing diagram of the DRAM read operation of an uncharged cells is shown in Figure 2.2. In order to reduce the electric field stress on the capacitor, one of the plates of the capacitor is usually biased to $\frac{V_{DD}}{2}$. Before the reading operation, the signal to precharge the bit lines (PEQ) is disabled. In normal operation, before reading the cell, the bitlines (BL and BLB) and sensing nodes (SA and SAB) are precharged to $\frac{V_{DD}}{2}$, and when the wordline is activated, the bitlines voltage will change slightly depending on the capacitance

of the storage capacitor. This slight change is detected by the sense amplifier as a '1' (Vdd) or '0' (Vss) as shown in Figure 2.2. In other words, the level of BL and BLB nodes eventually reaches the operating voltage (Vdd) or ground (Vss), respectively [45]. At startup, however, the storage capacitor has neither been charged to V_{DD} nor discharged to ground. Thus, at startup, the nominal voltage of each capacitor (Vc) is equal to the bias voltage $\frac{V_{DD}}{2}$ which is equal to the bitline precharge voltage. Thus, when read, the sense amplifier is equally likely to read a '1' or '0'. However, because of manufacturing variations, the storage capacitance of each bit will have slight differences, which leads to biasing of each bit to either a '1' or a '0'. This behavior is what allows the startup values of the DRAM to function as a PUF.

2.3.4 Uniformity of Memory

We start with an examination of the uniformity of a DRAM-based PUF. Ideally, 50% of the bits should be '1' and 50% should be '0'. For each of the 8 DRAMs used for startup value experiments, we took 10 measurements of the uniformity - i.e. the percentage of bits that were '1' or '0' at startup. As shown in Figure 2.3, without any write operation to the DRAM cells, they have start up values and some of the cells values are one. While not perfectly uniform, the uniformity is close enough to ideal that with proper bit selection it can be used as a PUF. This

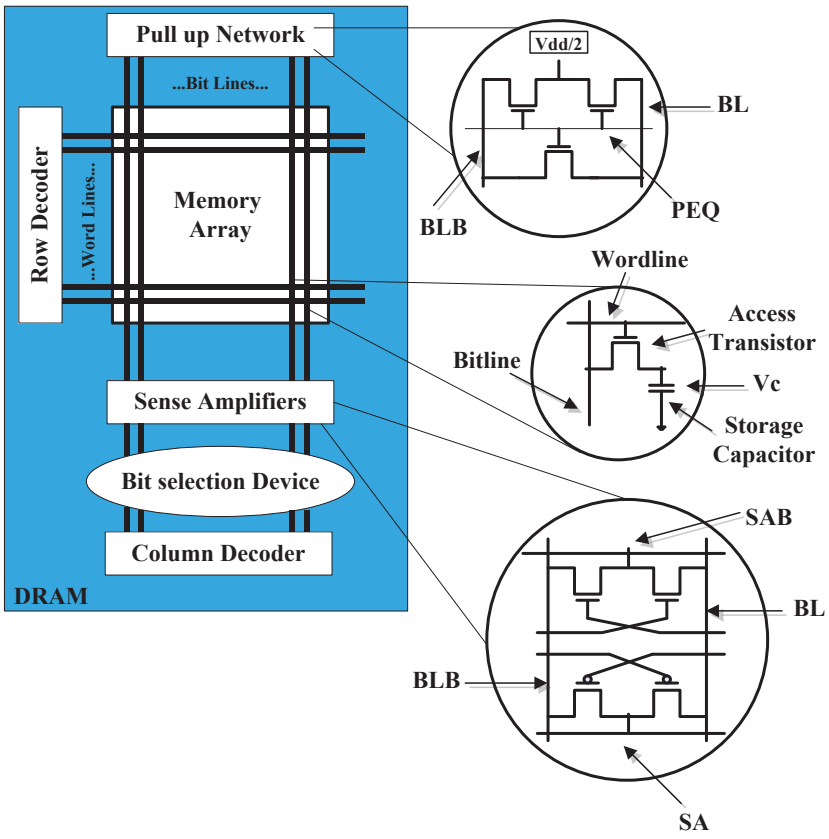


Fig. 2.1: Memory structure of a One-Transistor DRAM array.

error bar shows the average, minimum and maximum percentage of '1' values of each DRAM (DRAM1 to DRAM8) across different trials. As an example for DRAM1, the average, minimum and maximum percentages of 1's among all 10 measurements are 53.35%, 50.73% and 56.78%, respectively. As can be seen, there is a slight bias to '1' in all the DRAMs. Also, we looked at the distribution of 1's within some of the DRAMs (DRAM1, DRAM2 and DRAM3) to make sure that they are not accumulated in some parts and saw that they are uniformly distributed across all the DRAM cells.

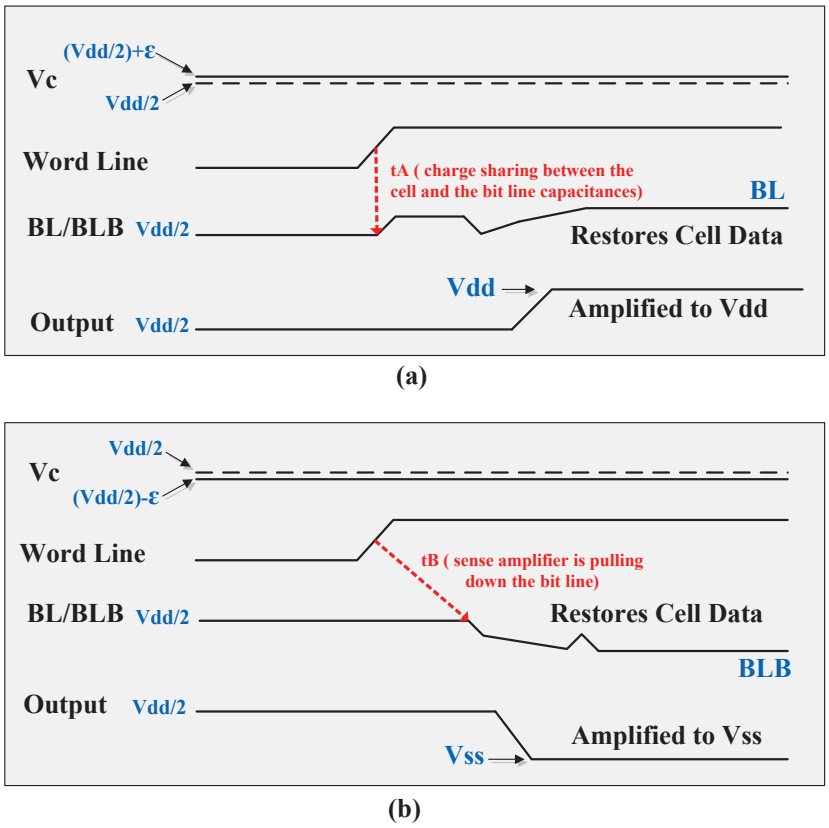


Fig. 2.2: Timing diagram of a DRAM read operation of an uncharged cells biased to Vdd (a) or Vss (b) due to process variations.

2.4 Experimental Setup

We used a set of 1-MBit HM51100AL CMOS DRAMs in DIP packages. Our setup essentially consists of four parts. Data acquisition experimental setup the FPGA based development board (Spartan 6 FPGA), the power supply & digital storage oscilloscope, the breadboard-based circuit (extension circuit), and the host PC. During data collection, the power supply supplies voltage to the extension circuit

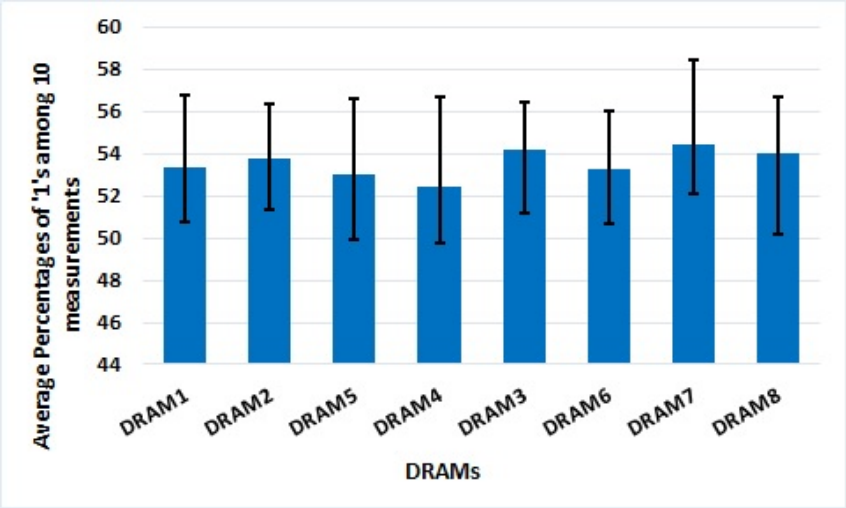


Fig. 2.3: Uniformity of DRAMs across 10 measurements for each of 8 DRAMs (average percentages of 1s values)

(off-chip DRAMs) that is mounted on the breadboard and we check the voltage levels using the oscilloscope. The communication between the host PC and the FPGA is composed of two connections: USB connection which is used for FPGA configuration download. The other one is a high density serial connector which is used for data communication between the PC software, ISE Design Suit 14.7, and the software running on the FPGA (Developed using Xilinx EDK). In other words, The FPGA was programmed to control the test sequence supplied to the DRAM chip and transmit the outputs of the DRAM to a computer using an on-board USB-UART module.

2.5 DRAM PUF Evaluation under Different Environmental Operating Conditions

Here, we examine the stability of the DRAM PUF bits under various environmental operating conditions. A stable bit is a bit that does not change in any trial and remains the same over different measurements of the same or different conditions. There are various parameters that can affect PUF stability such as process variation, PUF activity, temperature, supply voltage, etc. Others have proposed PUFs that take into account both process and environmental variations such as crosstalk which magnifies chip-to-chip signature randomness and uniqueness [100]. One of the advantages of our work is the stability evaluation against different operating conditions for more than one DRAM. We did all the experiments for three DRAMs which we will call them DRAM1, DRAM2 and DRAM3. We explore the differences between reliable and unreliable DRAM cell values, and the impact of operating conditions on them. To make a PUF highly reliable across its lifetime, unstable bits that are easily flipped by different operating conditions should not be used.

We start with baseline measurements (*Nominal Condition (NC)*) with the temperature set to 25 °C and the voltage to 5 V. For each DRAM, we took 10 measurements whereby we read all 1,048,576 (2^{20}) startup bits. The result from Table 2.1 shows that for DRAM1, 37.9% of the startup values were read as '0'

across all 10 measurements and likewise 43.5% were read as '1' across all 10 measurements. Thus, 81.4% of the bits are marked stable, and the remaining 18.6% of bits, which read as both '0' and '1' on different measurements, are marked as unstable.

Table 2.1: DRAMs Stability across Different Nominal Conditions

DRAM1		DRAM2		DRAM3	
Bit Value		Bit Value		Bit Value	
0	1	0	1	0	1
37.9%	43.5%	27.9%	37.6%	26.6%	37.7%
81.4%		65.5%		64.3%	

2.5.1 Stability of DRAM under Temperature Variation

We performed the experiments by sweeping the temperature from 0 °C to 80 °C using a ThemoStream system which is shown in Figure 2.4. In Table 2.2, we show the bit stability under both high temperature (80 °C) and low temperature (0 °C) conditions. NC-HT and NC-LT compare the stability of the DRAMs data under high temperature (HT) and low temperature (LT) conditions to the nominal condition (NC) stability, respectively - i.e. the percentage of nominal stable bits. In Table 2.2, stability means comparing a bit against the nominal condition. First, we derive the stable bits among the 10 measurements for each condition - i.e.

remain the same across all 10 measurements. Second, we find the stable bits among 10 measurements of the nominal condition. Finally, we identify which bits are stable across both sets of bits and the output provides the final stability results shown in Table 2.2.

Table 2.2: Stability under Temperature Variations compared to Nominal Conditions

	DRAM1	DRAM2	DRAM3
	% of stable bits	% of stable bits	% of stable bits
NC-HT	78.8%	64.4%	49.8%
NC-LT	49.9%	54.4%	44.3%

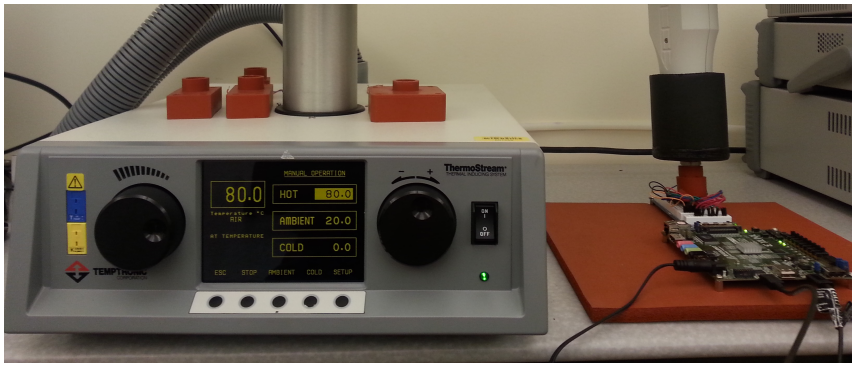


Fig. 2.4: Experimental setup with Xilinx Spartan-6 FPGA (right side) under the test using the ThermoStream system for high and low temperature variations.

Table 2.3: Stability under Voltage Variations compared to Nominal Conditions

	DRAM1	DRAM2	DRAM3
	% of stable bits	% of stable bits	% of stable bits
NC-HV	55.4%	54.3%	30.5%
NC-LV	43.3%	26.7%	23.8%

2.5.2 Stability of DRAM PUF under Voltage Variation

We vary the nominal supply by 10% up and 10% down, and observe the PUF's stability. Twenty measurements of startup values are taken at low voltage (4.5v) and high voltage (5.5v) (10 from each voltage). Table 2.3 contains the bit stability under both high and low voltage conditions for different DRAMs. Again, NC-HV and NC-LV compare the stability of the DRAMs data under high voltage (HV) and low temperature (LV) conditions to the nominal condition (NC) stability, respectively. As with temperature, we see that voltage variations can have an impact on the bit stability. The reason is because of the structure of a DRAM cell which consists of capacitor and a transistor. The startup values of a DRAM are dependent on the bias voltage of the capacitor, very slight variations in the power supply voltage can alter the voltage differential across the capacitor.

2.5.3 Stability of DRAM PUF due to Aging

Finally, we explore the potential impact of aging on the stability of the DRAM PUF. Several aging mechanisms can affect reliability during the lifetime of an IC. VLSI phenomena such as bias temperature instability (BTI), hot carrier injection (HCI), electro-migration and temperature-dependent dielectric breakdown (TDDB) are some of the causes of aging. As was mentioned in [101], among the BTIs, negative BTI (NBTI) affecting pMOS has more donating aging effect compared to positive BTI (PBTI) affecting nMOS. NBTI is enhanced by high temperature and high supply voltage. They both increase the threshold voltage and decrease the speed of CMOS transistors. A high switching rate in a circuit as well as excess supply voltage can enhance the HCI effect. A high operating voltage as well as higher temperatures can accelerate time-dependent dielectric breakdown (TDDB), a failure mechanism in MOSFETs.

In order to test the effects of aging on these DRAMs, we accelerated the aging process by performing burn-in of the DRAM using the ThermoStream burn-in system. We did 8 hours of high-temperature aging at 80 °C to approximate the effects of 6 months of aging. In Table IV, NC-AA refers to the aging condition, and the table compares the stability of the aged DRAMs data to the nominal condition stability. The amount of stability degradation is not constant for each device. Table 2.4 shows that after aging still 71%, 65%, and 55.1% of the cells

remain stable after aging across different measurements for DRAM1, DRAM2, and DRAM3, respectively.

Table 2.4: Stability under Aging Condition compared to Nominal Conditions

	DRAM1	DRAM2	DRAM3
	% of stable bits	% of stable bits	% of stable bits
NC-AA	71%	65%	55.1%

2.6 Bit Selection Algorithm for Generating PUF IDs

Bit selection algorithm is an algorithm to select a set of bits for an ID/key that has a high likelihood of being stable [106]. The key insight of the algorithm is that we use spatial information within the DRAMs to infer the stability of a bit cell. We have a grid for memory rows and columns that can give us a very good picture of the cell distribution in the memory array. Thus, spatial correlations (neighborhood stable cells) can be made in both x and y directions. In all, the algorithm uses spatial information within the DRAMs to infer the stability of a bit cell. In other words, stable neighbors provides better reliability than random selection. The basic algorithm is shown in Algorithm 1. The DRAM is organized as an array of cells - in our case for a 1-MBit DRAM the array is 1024 rows by 1024 columns. We count the number of stable bits (ones and zeros) in each row and then select rows that have more stable bits than specific thresholds (T1 and

T2). Thresholds have been chosen based on experimentation in order to select 2048 bits (16 128-bits keys). In the ideal case, half will be zeros and half ones, among 1 MBit data in the next level of the algorithm. T1 and T2 are different in order to get the equal number of bits (ones and zeros) for the PUF ID bits. In fact, we have an algorithm to adjust the thresholds (T1 and T2). First we selected a random threshold value and then based on the number of 1s and 0s that have been selected, our algorithm can change the threshold value to upper or lower value in order to get 50% of '1's (1024 bits) and 50% of '0's (1024 bits). T1 and T2 are the thresholds for choosing rows that have more stable '1's and '0's, respectively. Among selected rows, those bits that have stable neighbors also identified as potential highly stable bits suitable for enrollment as the PUF ID/key. Note that T1 and T2 should be selected in a way to find almost equal number of '1's and '0's bits for the IDs.

As shown in Figure 2.5, the most stable bits have been selected considering the neighboring cells. Basically, it shows how to select the most stable and reliable bits from 1024 rows by 1024 columns grid for PUF ID considering neighborhood cell stability approach. The more the number of neighbors are stable around a cell, the more reliable the cell is. In Figure 2.5, the green cell at the center with value 1 has the best chance to be used as an ID bit since it has 8 neighbors which are all stable. Similarly, the white cell is not very suitable to be used as an ID bit

as it has several unstable neighbors around. We considered a 1024 by 1024 grid (rows and columns) on the memory cells. PUF ID bits have been selected from the bits that are stable among all the constraints (different operation conditions).

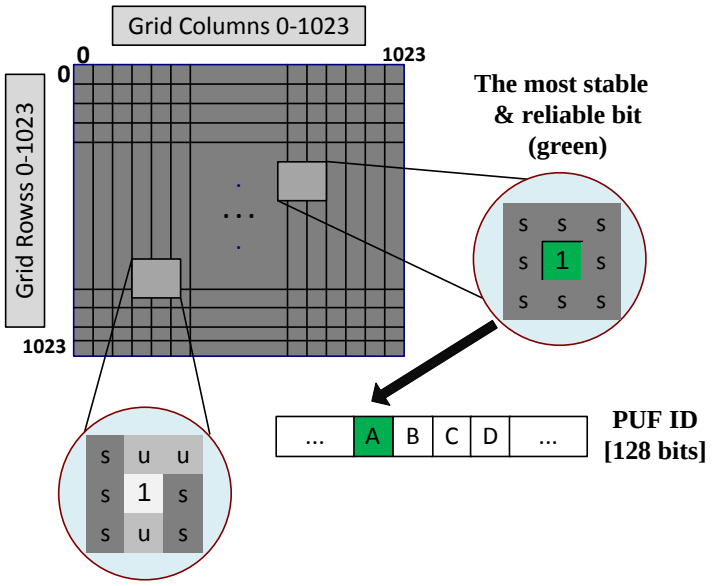


Fig. 2.5: Schematic of grids (rows and columns) on DRAM cells.

2.7 Analysis of Experimental Results and Their Validation

In the section, we first explain the multi-device evaluation on DRAMs based on different set of measurements, and then discuss the security metrics of uniqueness, randomness, and reliability followed by their results.

Algorithm 1 Highly Stable Bit Selection Algorithm for Selecting IDs from Different DRAM PUFs

1: Apply n measurements to each operating and environmental conditions including Normal Conditions (NC), High Temperature (HT), Low Temperature (LT), High Voltage (HV), Low Voltage (LV).

2: Find bits that are stable across all n measurements of all conditions for the specific DRAM that has been enrolled (DRAM1 or DRAM2 or DRAM3). Note that we do not perform aging during enrollment because it is not practical to age the chip because of the time involved.

3: Count the number of stable bits (ones and zeros separately) in each DRAM cells row.

4: Select rows R that have more stable bits than selected thresholds ($T1$ for '1's and $T2$ for '0's) which have been selected based on experimentation in order to select 2048 (16 IDs) bits.

5: For each row $r \in R$, enroll bits (r, j) that have a neighborhood of stable bits where r is the row number and j is the column number. The neighborhood of stable bits is defined such that row $r - 1 \in R$ and $r + 1 \in R$ and bits $(r, j - 1)$ and $(r, j + 1)$ are also stable.

2.7.1 Multi-device Evaluation on DRAMs

Here, we compare the results between DRAM1, DRAM2 and DRAM3. For each condition, we did 2-10 tests, collected data and the percentage of the stability among those data (for example for the high temperature condition, we had 2-10 different measurements). A bit is marked stable if it has the same result for all measurements. Figure 2.6 shows the percentages of 1s and 0s under different operational conditions and different set of measurements (2, 4, 6, 8, and 10) between DRAM1, DRAM2, and DRAM3. In most conditions, the degradation differences between the percentages of 2 measurements condition and 10 measurements are less than 10%. Figure 2.6 shows that by increasing the number of measurements, the percentage of stable bits, 1s and 0s (both of them) does not decrease very fast. Thus, most bits remain stable across multiple measurements.

2.7.2 Reliability

Reliability is a measure of repeatability or consistency with which a PUF generates its response across environmental variations, temperature, voltage, and aging. In our work, we chose stable bits based on the random selection and the proposed highly stable bit selection algorithm as discussed in the previous section. Various measurements from different operational conditions (HT, LT, etc) were used for each DRAM to apply bit selection algorithm on them. Then based on the

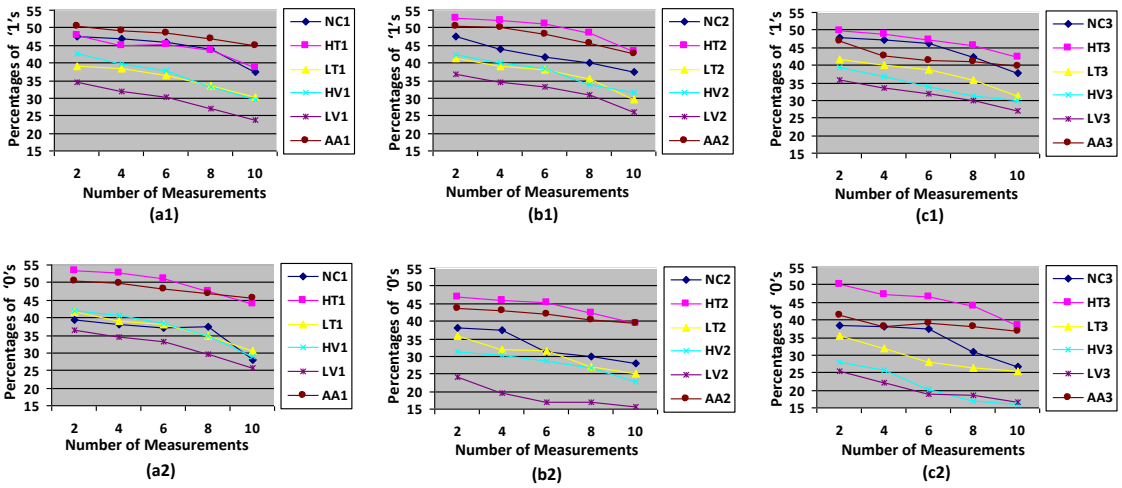


Fig. 2.6: Muti-device evaluation; (a1) & (a2), (b1) & (b2), and (c1) & (c2) show the stability (percentages of '1's and '0's) across different set of measurements for DRAM1, DRAM2, DRAM3 respectively.

number of distinct measurement approach (n=1 or n=2), we can determine which approach produced fewer bit flips during reconstruction phase. Our results show clearly that there is a relationship between better stability with bit selection and a higher number of distinct measurements (n), as shown in Table 2.5 and Table 2.6. We use Hamming Distance (HD) across different PUF measurements as the basis of our metric. To estimate the reliability metric, an n-bit response (R_i) from challenge C and from chip i should be extracted at normal operating condition (room temperature and normal supply voltage). The same challenge C is applied to chip i at a different operating condition to extract an n-bit response ($R_{i,2}$).

In the same way, T samples can be collected from chip i at different operating conditions. Hence, the average reliability metric (r) is estimated as:

$$r_i = \frac{1}{T}\sum_{t=1}^{T}\frac{HD(R_i, R_{i,t})}{n} \times 100\% \quad [63] \tag{2.1}$$

where $R_{i,t}$ is the t-th sample of R_i . The reliability metric shows the average number of reliable PUF responses. Ideally, this value should be 0.

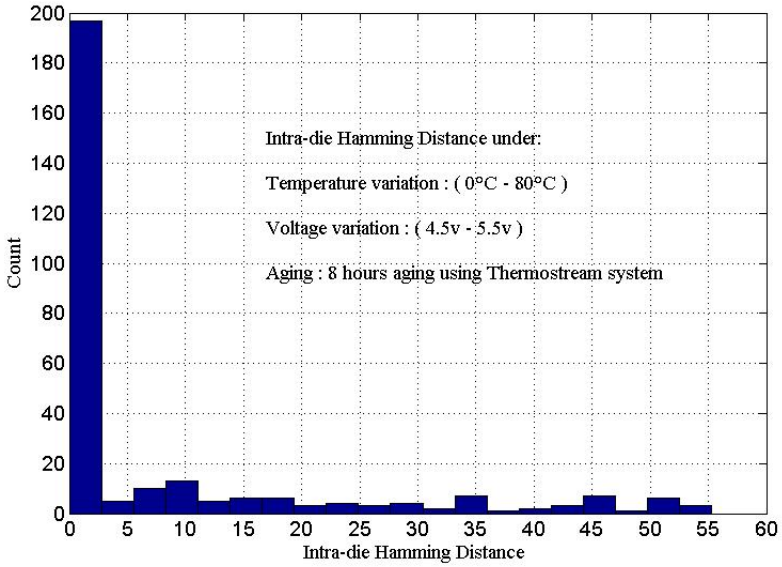


Fig. 2.7: Distribution of Intra-die Hamming Distance (HD) among 48 (3×16) DRAM-based PUFs under different operating conditions.

For measuring intra-die HD, we consider 48 IDs (16 IDs associate with each DRAMs). Each ID has been compared with different measurements of every operating conditions such as (Nominal Condition, High Temperature, Low Temper-

Table 2.5: Percentage of bit flips across multiple measurements (10-Sets) under Baseline (Base), Neighbor Selection (NS), Environmental Screening (ES), and combined (Algo) approaches.

	DRAM1				**DRAM2**				**DRAM3**			
	Base	NS	ES	Algo	Base	NS	ES	Algo	Base	NS	ES	Algo
n=1	78.5%	13.5%	19.8%	13.8%	78.3%	20.6%	15.4%	13.5%	75.6%	18.3%	21.7%	10.4%
n=2	76.1%	8.7%	11.3%	3.8%	72.5%	13.5%	7.1%	7.7%	72.2%	14.4%	9.9%	8.8%
n=3	73.9%	6.6%	6.9%	2.3%	71.8%	7.6%	4.7%	3.1%	71.9%	8.5%	7.5%	3.8%

ature, High Voltage, Low Voltage, and Aging). Figure 2.7 shows the distribution of intra-die HD of 48 IDs from 3 DRAMs under various conditions. As it is shown, most of the IDs are stable under different conditions.

Table 2.6: Percentage of bit flips across multiple measurements under normal operating conditions.

	DRAM1		DRAM2		DRAM3	
	BLine	Algo	BLine	Algo	BLine	Algo
n=1	47.3%	2.1%	50.4%	2.3%	49.5%	3.1%
n=2	45.9%	1.5%	46.9%	1.4%	44.2%	1.7%
n=3	45.2%	0.9%	45.7%	0.8%	43.1%	1.1%

2.7.3 Our Selection Algorithm vs Baseline Algorithm

As indicated earlier, we used our selection algorithm to select 16 128-bit keys from the available bits and compared it to a naive baseline algorithm where 16 keys are selected at random from the 1-MBit set. In both cases, the enrollment is done based on either one, two, or three distinct measurements. In other words, n = 1, n = 2, or n = 3 for the algorithm in Algorithm 1. The baseline case only uses 1, 2, or 3 nominal case measurements, whereas our selection algorithm uses 5-15 measurements - 1, 2 or 3 each for NC, LV, HV

2.7.4 Uniqueness

We evaluate the uniqueness of the DRAM PUFs. In particular, Uniqueness means that the responses resulting from evaluating the same challenge on different PUF instances should not be similar. The uniqueness of a PUF circuit among a population of PUF circuits manufactured, depends on various factors such as the process variation of a particular manufacturing process, any manufacturing defects and the metric used to evaluate uniqueness. Inter-die HD can be used to evaluate the uniqueness of the PUFs data. It is typically used which averages the hamming distance between responses of various PUFs over multiple CRPs. Assume that there are k-chips and R_i and R_j are the n-bit responses to a challenge C from chip i and j, respectively. Then the Inter-die HD among k chips is defined as: LT,

and HT. Aging is not used for enrollment because of the time involved and since it also shortens the lifetime of the device. Since our algorithm is a combination of a neighborhood selection algorithm as well as screening due to environmental measurements, we also evaluated each of these approaches separately. We evaluate the effectiveness of the selection algorithm by comparing the effect on reconstruction of the 16 keys. Ideally, on reconstruction we should read back the same bits. Table VI contains a summary of our PUF ID reconstruction results. Reconstruction consists of reading the keys back ten times under all conditions (NC, LV, HV, LT, HT and aging) - 60 reconstructions for each of the cases in the table. The data shows the number of bits that ipped in any of the reconstructions. As can be seen, the use of the enrollment algorithm (Algo) with just n = 1 - i.e. 5 measurements - reduces the number of bit ips for DRAM1 from nearly 79% in average for the baseline (Base) to less than 14%, which is sufficient for using the PUF for chip identification. In fact, our results show that we can also use this PUF for key generation with minimal ECC check bits. Furthermore, using more measurements during enrollment can decrease the number of bit ips significantly to 2-3%. It is interesting to note that while Neighborhood Selection (NS) and Environmental Selection (ES) are somewhat effective on their own, we get significantly better performance when both are used together (Algo). Note that Table VI is a worst case in that we examined the number of bit ips across multiple

measurements including under extreme operating conditions (40 measurements). Typically, however, reconstruction will be done under normal operating conditions. Table VII indicates the percentage of bit ips when reconstructing under just normal conditions. As can be seen, the bit ip rate is reduced to less than 3% with one set of measurements and to less than 1% with 3 sets of measurements.

$$Inter - dieHD = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{HD(R_i, R_j)}{n} \times 100\% \tag{2.2}$$

Ideally, the Hamming Distance (HD) between the responses should differ 50% of total responses bits.

We calculated the average Inter-Hamming distance between all pairs of IDs that were extracted from the different PUFs (DRAM1, DRAM2 and DRAM3) based on our bit selection algorithm. Figure 2.8 shows the distribution of Inter-die HD of the 48 IDs from the 3 DRAMs. The average HD is 0.4937 and close to the ideal 0.5. Hence, the proposed DRAM PUFs can provide unique identifiers. As shown in Figure 2.8, the HD points tend to be very close to the mean of the set, as can be seen by the very small standard deviation of 0.055.

2.7.5 Randomness

In PUF design, the randomness of the data is very important as it can prevent the prediction of the cell values or the ID bits. In other words, perfectly random data

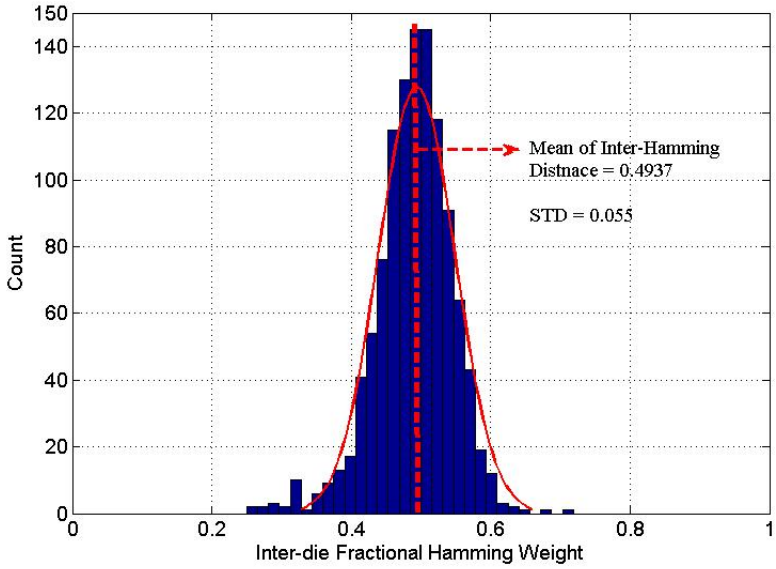


Fig. 2.8: Distribution of Inter-die Hamming Distance (HD) of 3 DRAMs among Different the extracted IDs.

means that the PUF cells are generated independently of each other, and the value of the next cell cannot be predicted, regardless of how many cells have already been produced. PUFs using intrinsic randomness are very attractive as they can be included in a design without applying any modifications to the manufacturing process. Note that if the HD uniqueness measure discussed above is 50%, it does not mean that data is necessarily random. To evaluate the Randomness of a PUF, statistical tests such as the NIST Test [87], Machine Learning (ML) techniques, Shannon entropy, or Min-entropy can be applied to the PUF data. Here, we considered Min-entropy as a metric to estimate the unpredictability (randomness) of our DRAM PUF data.

Min-entropy is an approach for estimating the randomness of the PUF responses based on experimental data [54]. In particular, min-entropy indicates how many bits of a PUF response are uniformly random. In this literature, we estimate the entropy and min-entropy of the responses of all available PUFs. We have three DRAMs and for each of them, 16 IDs were selected based on the algorithm. Min-entropy is estimated as:

$$P_{MAX} = MAX\{Hwt(i), 1 - Hwt(i)\} \tag{2.3}$$

$$\text{Min-Entropy} = \frac{1}{128}\sum_{i=1}^{128}(-\log_2(P_{MAX}(i))) \tag{2.4}$$

where i is the number of ID bits. First, we have to consider Hwt(i) for each bit over all IDs. In fact, the Hamming weight of a bit Hwt(i) is defined as the number of non-zero bits. The min-entropy that has been calculated based on Equation 1 which is 0.9483. This value is approximately close to the ideal case min-entropy of 1.

2.7.6 Case Studies

We evaluate security analysis such as reliability, uniqueness, and randomness for two different cases for ID extraction.

- **Case1:** Here, our goal is to find the maximum number of IDs using Algorithm 1. The enrollment and reconstruction are exactly similar to the

results in Table 2.5 and Table 2.6 except the thresholds (T1 and T2) have been set to zero.

- **Case2:** In this case, we try to find the maximum number of IDs with 100% reliability. To achieve this goal, 40 measurements from different operating conditions have been considered to extract the bits that are stable. Then we apply Algorithm 1 on the stable bits to directly select the ID bits. In other words, for the enrollment phase, all measurements of all conditions (40 measurements) are considered instead of one measurement of each conditions (5 measurements). Similar to Case1, the difference between the result from Case2 in Table refcases12 and Table 2.5 is that the thresholds (T1 and T2) in Algorithm 1 are zero for Case2. Note that in Table refcases12, Reli. is the Reliability, Unique. is the Uniqueness and Min-Ent. is the Min-Entropy.

From Table 2.7, the maximum number of IDs for DRAM1, DRAM2, and DRAM3 are 263, 178, and 205 in Case1, and 16, 29, and 22 in Case2. As is shown in Table 2.7, reliability, uniqueness, and min-entropy have been calculated in different cases for different DRAMs using equations 1, 2, and 4 respectively. As a result of using all measurements for Case2, we are able to get 100% reliability, better uniqueness metrics, and a min-entropy that is very close to ideal. However, we are not able to enroll as many IDs in Case2 because of the restrictive enrollment process. As can be seen, the reliability results in Table 2.7 are lower than what

was shown in Table 2.5 and Table 2.6.

Table 2.7: Quality Evaluation of IDs from DRAM PUFs.

	DRAM1		DRAM2		DRAM3	
	Case1	**Case2**	**Case1**	**Case2**	**Case1**	**Case2**
No. IDs	263	16	178	29	205	22
Reli.	81%	100%	75.4%	100%	70.7%	100%
Unique.	0.539	0.503	0.581	0.514	0.448	0.492
Min_Ent.	0.89	0.99	0.78	0.96	0.86	0.98

2.8 DRAM PUFs Reliability Analysis due to Device Accelerated Aging

2.8.1 PUFs under Accelerated Aging

Many PUFs are known to suffer from aging and lose their reliability over time - i.e. they no longer consistently return the same responses. As reliability goes down, the PUF loses its usefulness as a practical authentication device. The aging effects on a PUF are due to the degradation of transistors as a consequence of aggressive scaling in CMOS in recent years [46] [80]. As technology has entered the nanometer regime, several important factors cause degradation in transistor including negative bias temperature instability (NBTI) [84], hot carrier injection (HCI), and temperature dependent dielectric breakdown (TDDB) [70]. This

degradation mainly manifests itself as an increase of transistor threshold voltage. NBTI causes a gradual increase in the threshold voltage and decrease in the mobility, drain current and transconductance, which is most evident and occurs in switched-on PMOS transistors. Recently, there has been significant work that evaluated the effects of environmental conditions especially accelerated aging on different types of PUFs. Abhranil et al. performed an accelerated aging testing on an FPGA-based RO PUF and analyzed its affects on the functionality of the PUF in [62]. Based on their observation, aging makes PUF responses unreliable but the randomness of PUF responses remains unaffected. Wei et al. proposed a method that can tolerate bit errors from responses of PUFs to make them highly reliable in [110] and [109]. In [28], Achiranshu et al. evaluated the device aging effect to present a technique for improving uniformity (distribution of 1s & 0s) and reliability. Another study has been done by Dinesh et al. in [27] on evaluating device aging on the stability of ring oscillator PUFs for different PUF circuit-level choices and operating conditions. In [59], the authors investigated the effects of data-dependent silicon aging on SRAM PUF reliability under a number of realistic scenarios. They claimed that some scenarios even show anti-aging effects. In 2016, Rahman et al. [78] proposed a new aging-resistant design that reduces sensitivity to negative-bias temperature instability and hot-carrier injection stresses. Since DRAM PUF behavior is primarily determined by process variations in the

storage capacitor rather than variations in the transistor, it is likely that aging due to NBTI may not be an issue. In this section, we investigate the effects of aging on DRAM PUFs functionality. Specifically, we have the following contributions. 1). Study of the impact of Negative Bias Temperature Instability (NBTI) on the read stability of DRAM PUFs over a period of 18 months. 2). Investigation of the reliability of aged DRAM PUFs selected IDs. 3). Aging analysis using implementation of realistic aged DRAM chips.

Transistor-based Aging Effects

Aging is an extremely important issue for devices that must remain in operation for decades. Transistor aging effects in nanoscale CMOS result in transistor performance degradation over the device life-time. The primary physical mechanism behind transistor aging is Bias Temperature Instability (BTI). Such transistor aging results in circuit performance degradation over time. Bias Temperature Instability (BTI) effect can be further divided into NBTI and PBTI. NBTI effect degrades the performance of PMOS over time by increasing its threshold voltage. Similarly PBTI effect degrades the performance of NMOS. As the use of High K (dielectric constant) materials is increasing, PBTI effects are also increasing. Moreover, due to transistor aging effects in nano-scale, circuit delay will increase over the lifetime the device. Transistor aging also impacts the clock skew over

time [85]. In the case of a DRAM PUF, however, transistor aging due to BTI does not affect the behavior of the PUF. Since the transistor is only used as an access transistor, changes in the threshold voltage or speed of the transistor do not change the startup value of the DRAM cell.

Capacitor-based Aging Effects

Over the last decade, significant increases in MOS capacitor reliability have been achieved through a combination of advanced manufacturing techniques, new materials, and diagnostic methodologies to provide requisite life-cycle reliability for high energy pulse applications. In general, the capacitance of certain capacitors decreases as the component ages. In more details, the relationship between life/aging rate and the dielectric in a capacitor is usually expressed in terms of a power law where the change in life will be equal to the inverse of the change in stress raised to some power [89]. For the first time, a capacitor technology has been developed in [89] in 1997 that is essentially graceful-aging and contains no intrinsic single point failure modes. Within the context of a DRAM PUF, changes in the capacitance due to aging do not affect the start up value of the DRAM cell, since the determining factor of the start up value is variations in the bias voltage of the capacitor.

2.8.2 Experimental Platforms

Our proposed approaches are evaluated on an experiment platform, the Xilinx Spartan 6 FPGA on a Digilent Atlys board. The off-chip DIP DRAMs were mounted and wired to a prototype board that has a high density serial connector during data collection. The serial connector allows the prototype board to interface with the FPGA. Then, programmed FPGA controls the test sequences applied to the DRAM and transmit the results/outputs from the DRAM chip to a computer (workstation) using a USB-UART module. All experiments are performed using our ThermoStream Burn-in System (Temptronic TP04100A ThermoStream Thermal Inducting System) to accelerate aging. Each measurement is performed at a VDD of 5V and at room temperature and consists of 144 start up state readings at different dates from Sep. 2014 to Feb. 2016.

2.8.3 DRAM PUFs Aging Effects Study and Procedure

in Section 2.5, we showed the results of different experiments on DRAM PUFs in different operating conditions i.e. high temperature, low voltage, aging, and etc. Here, we particularly focus on the effect of long-term aging on DRAM PUF behavior. We selected a 1 MBit HM51100AL CMOS DIP DRAM and evaluated 3 different DRAM ICs over the course of 18 months from Sep. 2014 to Feb. 2016. To accelerate the aging on 3 DRAMs (DRAM1, DRAM2, and DRAM3), in Sept.

2014 we initially performed burn-in of the DRAM using our Thermostream burn-in system. We did 8 hours of high-temperature aging at 80 °C on dram1, dram2 and dram3 to approximate the effects of 6 months of aging. For data collection, we need to get various measurements before and after aging the DRAMs. Before aging the DRAMs, we gathered different startup measurements at Nominal Condition (NC) (operating temperature: 25 °C and voltage: 5 volt). After the DRAMs were aged in Sept. 2014, we again obtained measurements from each DRAM in Sep. 2015, Feb. 2015, Mar. 2015, Apr. 2015, Jul. 2015, Aug. 2015, Jan. 2016, and Feb. 2016. Thus, in summary, that we took 10 measurements from each DRAM at nominal conditions in Sept. 2014 (30 measurements in total for all 3 DRAMs under NC), and 6 measurements from each DRAM at each date of the year from Sep. 2014 to Feb. 2016 (144 measurements in total for all 3 DRAMs after aging). After getting those measurements, we need to evaluate and analyze the stability of DRAM PUFs under nominal and aging conditions. A bit is defined as stable if it remains the same under different measurements. Table I shows the stability of dram1, dram2, and dram3 due to fresh (un-aged) and aged conditions at different dates of the experiment and compared with nominal conditions. For example, the percentages of stable bits ('1's and '0's) on Sep. 2014 at pre-aging condition are 88.9%, 91.6%, and 89.7% for dram1, dram2, and dram3, respectively. As shown in Table 2.8, at aged-DRAMs condition, the percentages of stable bits do

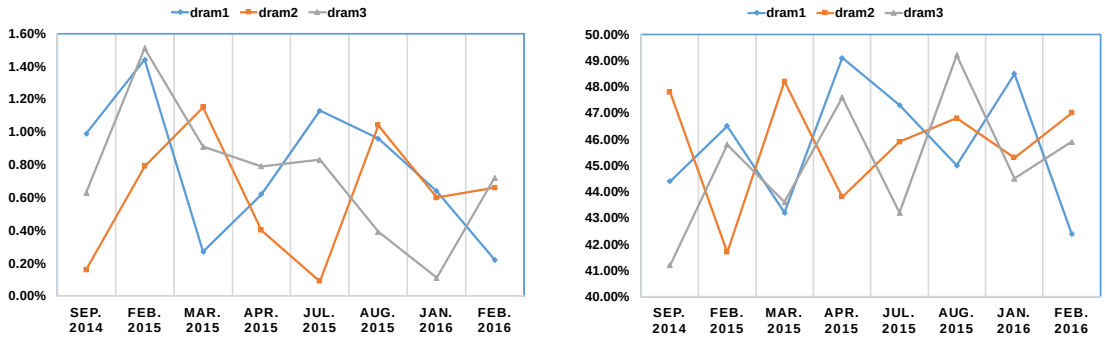
not change that much especially for dram1. Furthermore, these small changes are very normal and still a large amount of data are stable during the 18-month period of time of our experiments on DRAMs. Based on our observations, our DRAMs (dram1, dram2, and dram3) which are under experiments for aging effects on PUFs functionality, are much more stable than what we even expected before starting this project, since it is a common expectation that aging has irreversible effects on devices.

Table 2.8: Stability of DRAMs due to Aging compare to Nominal Conditions

	dram1	dram2	dram3
	stable bits (%)	stable bits (%)	stable bits (%)
Pre-aging (un-aged) Condition			
Sep. 2014	88.9%	91.6%	89.7%
Aged DRAMs			
Sep. 2014	87.1%	90.1%	80.2%
Feb. 2015	86.4%	85.1%	83.1%
Mar. 2015	90.2%	83.2%	78.0%
Apr. 2015	85.8%	82.4%	76.6%
Jul. 2015	87.3%	81.7%	81.3%
Aug. 2015	86.7%	81.2%	80.1%
Jan. 2016	86.2%	82.7%	82.4%
Feb. 2016	87.5%	84.6%	81.9%

2.8.4 Reliability of DRAM PUFs under Aging Condition

Here, we analyze and examine the security properties of the three aged DRAM PUF through reliability. Reliability gives a dimension to evaluate whether the PUF generates stable responses for the same challenge under different environ-



(a) (b)

Fig. 2.9: Percentage of bit flips across multiple measurements (a) under accelerated aging condition (Bit Selection Algorithm), (b) under random condition (Random Bit Selection)

mental conditions. In other words, reliability means that the secret key or ID generated by the DRAM PUF will be the same despite any changing operating conditions. Note that we select 2048 bits (16 128-bits keys) from each DRAM PUF and ideally half of the bits should be zeros and half ones. To ensure the reliability of the DRAM PUF response, the output needs to either be error corrected or analyzed in such a way that only stable cells are selected and used. Furthermore, we looked at DRAM cells arrays in a two dimensional (2D) structure including rows and columns, and 1 Mbit DRAM is a 2D image of 1024 rows by 1024 columns. Also, we have identified that neighbors stability status can be used to determine the most reliable DRAM cells for IDs which is a simple bit selection algorithm for selecting ID bits. In this algorithm, we try to find the cells that have the highest

number of stable neighbors around. Those bits that have more stable neighbors identified as potential highly stable bits which are suitable as the PUF ID/key. Indeed, a stable cell which is surrounded by more stable cells is more likely to remain stable because its neighboring cells have experienced similar aging effects.

Figure 2.9 a shows the percentages of bit flips across various measurements from Sep. 2014 to Feb. 2016 of dram1 (blue), dram2 (orange), and dram3 (gray) under aging condition after applying the bit selection algorithm. Note that reliability is obtained from Equation 2.5. As can be seen, the percentages of flip bits are changing over time for all three DRAMs. For example, bit flips for dram3 decreases significantly from Feb. 2015 to Jan. 2016 and increases to Feb. 2016. There are various fluctuations in Figure 2.9 a in terms of bit flips. In some cases, the reliability increases (bit flips decreases) and vice versa. In the worst case, we can see less than 1% increase in the percentage of bit flips in dram2 from Sep. 2014 (0.16%) to Mar. 2015 (1.15%). On the other hand, in the best case, the percentage of bit flips drops from Feb. 2015 to Mar. 2015 in dram1 from 1.44% to 0.27% (1.17%) which is a huge increase in the reliability. As shown in Figure 2.9 a, we can see more reliability improvement rather than reliability retrogression. It seems that the initial aging measurement was within the margin of error that we see over time, and one could make the case that aging has very little effect on permanent behavior of the PUF. On the other hand, Figure 2.9 b shows the re-

sults we got from random bit selection. There are huge differences between what we got out of our algorithm (bit selection algorithm) and random bit selection procedure. In some cases, almost 50% of the bits flip during random selections.

$$Reliability(\%) = 100\% - FlipBits(\%) \tag{2.5}$$

Conventionally, another metric to estimate the reliability of our DRAM PUFs under aging condition is represented by the (normalized) Intra-die Hamming Distance (HD) of the PUFs responses. An n-bit reference response (R_i) is extracted from the chip i at the nominal condition. The same n-bit response is extracted at a different operating condition (in particular, accelerated aging condition) with a value (R'_i). S samples of (R'_i) is taken for each of the operating conditions. The reliability is defined as follows which is the average Intra-die Hamming Distance:

$$\frac{1}{S}\sum_{t=1}^{S}\frac{HD(R_i, R'_{i,t})}{n} \times 100\% \tag{2.6}$$

where $R'_{i,t}$ is the t-th sample of R_i. The responses that being compared here are produced from the same DRAM chip.

In more details, for the calculation of conventional intra-die HDs, we use the readout response values after applying our bit selection algorithm for selecting ID bits, 16-128 bits IDs from each DRAM, regardless of whether the responses have stabilized or not. Based on the IDs collected from 3 DRAM PUF instances (dram1,

dram2, and dram3), with 48 IDs in total, we obtained the distribution of intra-die HDs shown in Figure 2.10.

Figure 2.10 displays the Hamming distance between different IDs extracted from each DRAM PUFs in order to evaluate their reliability. Because of the space limitation, we demonstrated the intra-die HD of 3 DRAMs at each date (i.e. Sep. 2014 in Figure 2.10(a)) in 1 histogram. We know that the intra-die HD is the hamming distance between the IDs/keys of an individual DRAM. The vast majority of Hamming distance values are between 0 and 0.5% for all the dates (Sep. 2014 to Feb. 2016) as shown in Figure 2.10. Based on the results we got from the intra-die HDs in Figure 2.10, we see that most of the IDs remain stable even after 18 months.

2.9 DRAM PUF as a System Security Solution

Since DRAMs are a system/board level component, they offer an opportunity to use the DRAM PUF to authenticate the system. However, since the DRAM is not embedded within another Integrated Circuit (IC), it is potentially subject to attacks that may compromise the PUF. Two main vulnerabilities are that the pins are easily probed and the DRAM can be removed/replaced from the board. Since the pins of the DRAM are easily accessible, an attacker could exhaustively read all the memory cells in the DRAM and store the startup values. Using these

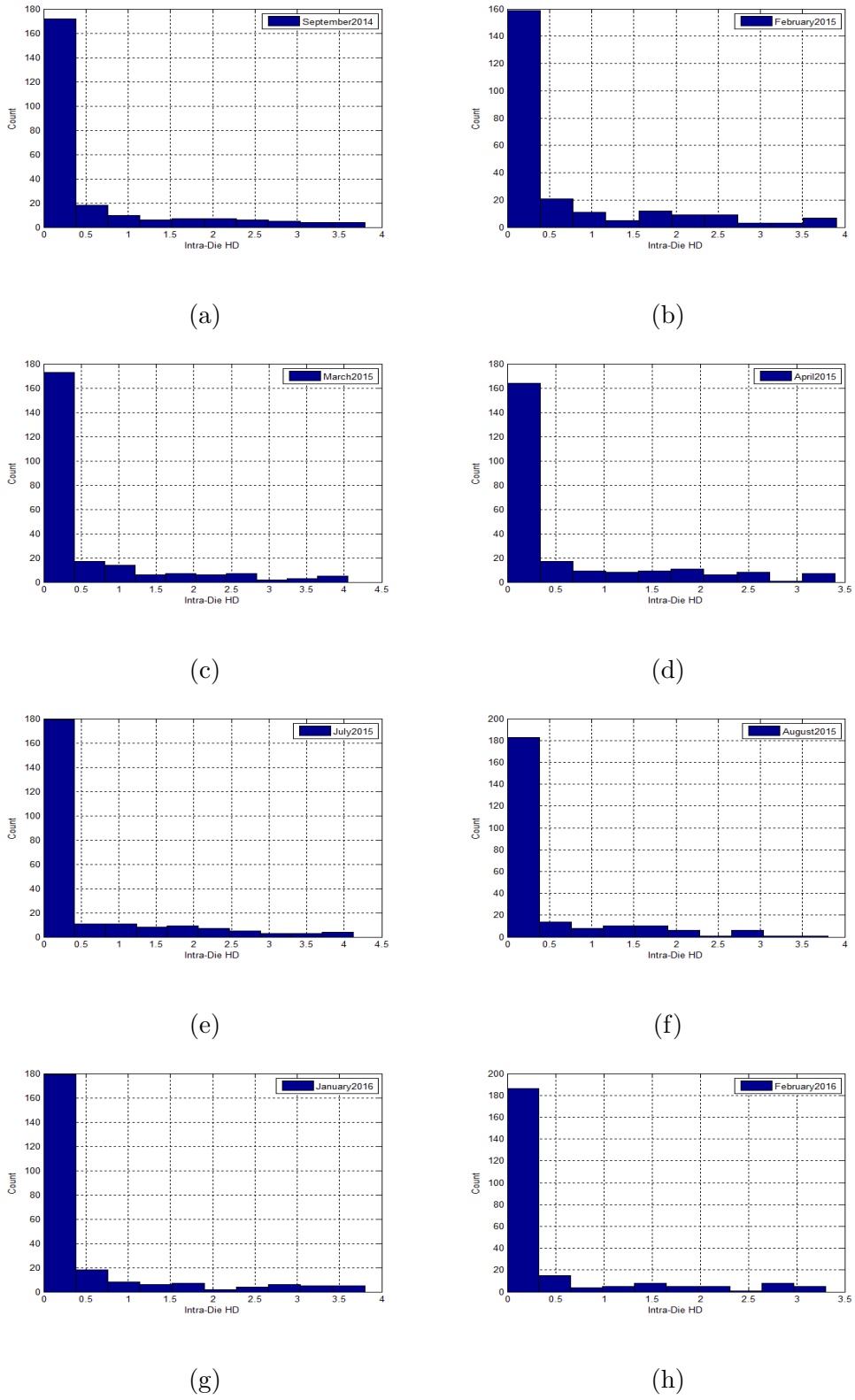


Fig. 2.10: Percentages of HDs of DRAMs among different IDs under Aging condition in (a) Sep. 2014, (b)Feb. 2015, (c) Mar. 2015, (d) Apr. 2015, (e) Jul. 2015 (f) Aug. 2015, (g) Jan. 2016, and (h) Feb. 2016.

stored values, the attacker could then theoretically reproduce the PUF responses of that DRAM. Note that, because of the difficulty of replicating process variation, it would be impossible to actually replicate a DRAM with the same physical responses. Instead, however, an attacker could add nonvolatile storage to the DRAM to store the startup values. Because of the large size of typical DRAMs, this attack is relatively impractical. Essentially, one would need to more than double the cost of the DRAM to add the nonvolatile storage. Moreover, the attack would need to determine whether the DRAM is in startup or not, meaning that there would need to be extra circuitry to detect writes to the DRAM. A simple approach would turn off the nonvolatile storage and turn on the actual DRAM on detection of the first write. That could easily be defeated by simply writing a few random cells at startup, thus requiring the attacker to monitor writes to every cell thus increasing the cost of the attack significantly. In addition, if suspicious of attacks, it would be relatively easy to determine that the DRAM package has an extra embedded nonvolatile storage component by imaging the DRAM package.

As to the second vulnerability, an attacker could remove the DRAM package from an authentic system and place it in another potentially counterfeit system and thus allow the second system to be authenticated as valid. While this is possible, it does not allow the attacker to create a new "authentic" system without having access to a previously valid system. Presumably the previously authen-

tic system is non-functional or has been taken out of the supply chain, because otherwise the cost of the authentic system would make the need to counterfeit it irrelevant. The attacker would need one authentic system for every counterfeit system. To address this problem, manufacturers must keep a tight control of their supply chain and ensure that any authentic systems that have been taken out of the supply chain must either destroy the DRAMs or simply remove the DRAM responses from the database.

2.10 Conclusion

This chapter identified unexpected startup behavior in a DRAM that could allow the DRAM to be used as a PUF primitive with proper bit selection to maintain high reliability and stability. We presented a novel PUF ID generation approach and evaluated the practicality of our PUF with empirical data that was obtained from a set of real DRAMs. Our experiments showed that temperature, voltage, and aging can have a major impact on DRAM PUF stability. We proposed a specific enrollment algorithm for generation of a stable PUF using memory cells within a DRAM module. The evaluation of DRAM gives insight into the randomness of the cells startup values, and their stability. The experimental results demonstrate that our algorithm is very effective at finding the most stable bits to be used as a 128-bit identifier. We also show that the DRAM PUFs randomness

and uniqueness metrics are close to ideal. While DRAM PUFs may not exhibit the same levels of stability as newer SRAM PUF techniques, they offer an opportunity for PUFs in systems that do not have SRAMs, require high numbers of PUF CRPs, or want higher density than available with SRAM.

Chapter 3

DRAM-based Random Number Generation Design

3.1 Robust Hardware True Random Number Generations using DRAM Remanence Effects

3.1.1 Introduction

In this chapter, we introduce a new approach for constructing a robust hardware TRNG based on DRAM remanence effects. A robust hardware TRNG should have a high level of randomness or unpredictability in the output. Otherwise, an adversary can simply query the generator and predict the sensitive data. Our novel approach focuses on creating a TRNG from DRAM devices or components that are already present in most computing systems. This feature effectively allows for DRAM to generate a TRNG without any extra hardware required to be added to the system. Using this knowledge we designed an inexpensive cryptographic key generator, implemented the findings and analyzed the results. The main advantage of this approach over other TRNGs is that we can produce highly random

bits without incurring any additional hardware costs. We also propose random number generation based on DRAM startup behavior which will be presented in Section 3.2.

3.1.2 Data Remanence

Data remanence is the residual information that remains on a storage medium even after erasure (data clearing) or powering off the device. Data remanence as a problem was first discovered in magnetic media [26], [34]. Thus, there is a possibility that sensitive information still can be extracted from non-volatile memory that had presumably been erased. Additionally, supposedly volatile memories, such as DRAM, are examples of storage components that they do not lose their data immediately after power is removed. Furthermore, these memories exhibit remanence effects wherein data stored for an appreciable amount of time in the same location can leave a permanent recoverable trace in the memory cells. Data remanence effects can play an important role in extracting the secret stored information from volatile memories. As mentioned in [26], data remanence is not a directly evaluated criterion of trusted computing systems, but it is critical to the safeguarding of information used by trusted computing systems. In [36], Halderman et al. have shown that DRAMs retain their contents for several seconds after power is lost.

3.1.3 DRAM Remanence-based TRNG

Here, we describe our methodology of using the DRAM remanence effect to propose a new TRNG model. We start with a brief review of a typical DRAM architecture. A DRAM memory cell uses a single transistor and a capacitor to store a bit of data. Cell information (voltage) is degraded mostly due to a junction leakage current at the storage node. Therefore, the cell data must be read and rewritten periodically even when memory arrays are not accessed. Essentially, the DRAM controller must refresh each cells voltage before it decays to the point where the bit information gets lost. Normally, the refresh rate is so high that each cell gets refreshed several times per second. Figure 3.1 shows a DRAM cell array that are arranged in rows (0 to n-1) and columns (0 to n-1) of memory cells called wordlines and bitlines, respectively. Each memory cell has a unique location or address defined by the intersection of a row and a column. One memory cell has been magnified in Figure 3.1 which consists of a access transistor and a storage capacitor that is charged to produce a 1 or a 0. The processes of extracting random bits from DRAMs, while considering the remanence effect and startup behavior of DRAMs is briefly laid out as follows. As shown in Figure 3.2, the process is first we write the value 1 to all cells of the available memory; this can be seen as step a) in Figure 3.2. After the write operation, a delay function (step b) has been applied to turn OFF the DRAM for certain amount of time, which is in

milliseconds, and then turn the DRAM back on after a specific delay time. In the next step, the entire 1 Mbit of data are read (step c) and stored.

The expectation was that, depending on the delay, a certain percentage of the 1s written at startup would have flipped to 0 and the remainder would have stayed 1 because of remanence. Presumably, which bits flipped would be random and that could provide the bits for our TRNG sequence. However, it turns out that DRAMs do not actually settle to all 0 when powered off completely. In Chapter 2, we showed that certain DRAMs exhibit behavior similar to static RAMs (SRAMs), i.e. they have seemingly random startup values after being powered on. Thus, for determining the effect of remanence after various delays, in our experiment we found that instead of flipping to '0', the bits will settle to the original startup value at power on.

Experimental Setup

We experimented with several delay values to determine the effect of remanence on the bit values. We used a Xilinx Spartan-6 XC6SLX45 FPGA experimental platform similar to [49]. A power-cycle circuit was implemented in our experimental setup as to provide a method to deliver power to the DRAM chip and as well as enabling (turning ON the DRAM) and disabling (turning OFF the DRAM) the on-board DRAM chip located on the Xilinx Spartan-6 FPGA board. We

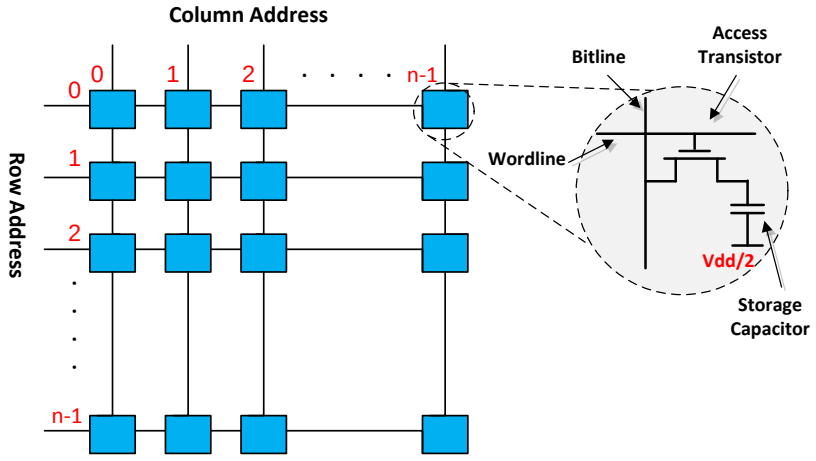


Fig. 3.1: DRAM cells array with a typical single MOSFET transistor and a storage capacitor.

collected approximately 400 measurements of DDR2 SDRAMs based on various delay time and under nominal environment conditions (temperature: 25 °CC and DRAM voltage: 1.8 volt).

Our TRNG Model

Figure 3.3 shows the results of our experiments where the x-axis shows the delay times from 1 milliseconds to 2000 milliseconds and the y-axis illustrates the percentages of 0 bits observed at each delay time. As can be seen, Figure 3.3 is divided in to three regions (Flip, Fluctuation, and Startup). Specifically, we note that before the DRAMs settle into their startup values where the data is quite stable, the DRAM values somewhat "overshoot" in that slightly more bits flip to 0

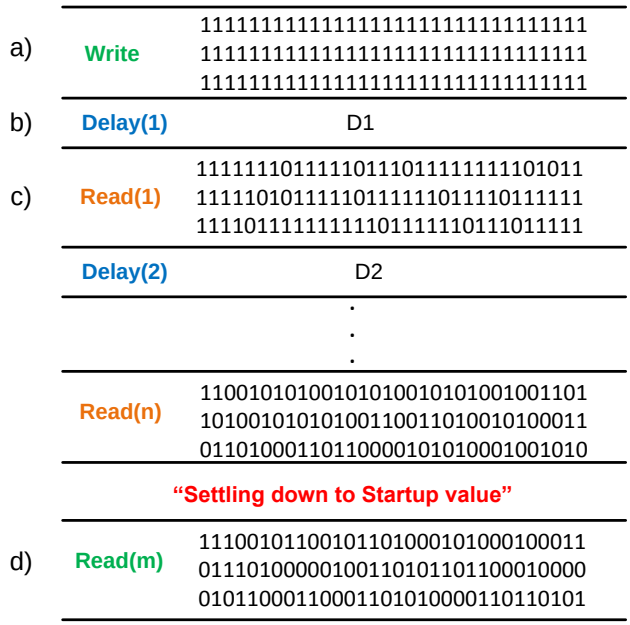


Fig. 3.2: Real data snippets that illustrate the DDR2 SDRAM operations: a) Write. b) Delay(1). c) Read(1). d) Read(m).

before flipping back to 1 in the startup state. We call this region the "fluctuation" region. In the following subsections, we will discuss and model the three distinct regions.

- **Flip Region:** During the flip Region, the percentage of flip bits increases from below 1% to 56% as the delay time goes from 1 milliseconds up to 350 milliseconds. We showed 5 measurements at each delay time, and showed the percentage of bit flip with an error bar. In total, we performed 175 measurements in this region. Using MATLAB cftool, we draw the fit data that is represented by the red line. The fit equation is a Linear model

Equation with the below specifications:

$$b_0 = 0.211t - 5.909 \tag{3.1}$$

where t is the delay time, and b_0 is the expected number of zero bits at each delay time within the Flip-Region.

- **Fluctuation Region:** The Fluctuation Region starts from 350 milliseconds and ends at 1180 milliseconds. This region has more variations and most of our measurements are focused on this region. In total, we took 168 measurements in this region. The percentage of bit flips ranges from 58% to 64% which is more than the percentage of '0's at startup. That indicates that there are more '0's in the fluctuation region than the startup. This unexpected phenomenon became the focus of our investigation. A potential cause for this unexpected behavior is the temperature effect that we will discuss more about it in Section 3.1.3. The expected data of this region is drawn by a red line and the fit equation is a Linear model as seen with Equation 3.2.

$$b_0 = 3.744e^{-5}t + 61.27 \tag{3.2}$$

- **(3). Startup Region:** Finally, the Startup Region is where the DRAM's behavior settles down to its expected startup behavior. This region is where

the delay goes from around 1200 milliseconds and beyond. The linear model of this region is as follows:

$$b_0 = 0.0002923t + 56.3 \tag{3.3}$$

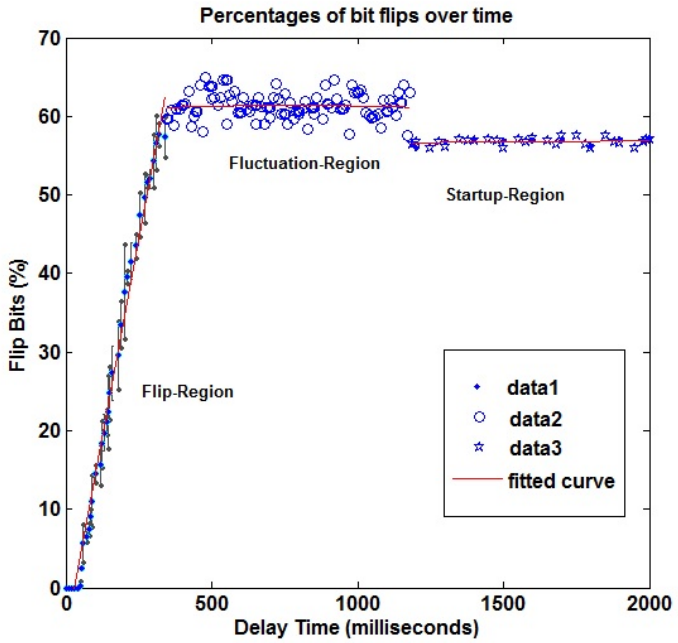


Fig. 3.3: Our DRAM Remanence based TRNG Model using MATLAB Curve Fitting Tool (CFTool).

Impact of Temperature on DRAM Remanence

Previous works have characterized DRAM and SRAMs memory remanence as a function of temperature. In fact, there is a correlation between DRAM remanence and DRAM temperature. Temperature plays an important role in data remanence

time and the amount of bit flips [83] by affects power leakage at the transistor level. Particularly in DRAM, it significantly decreased with increasing the operation temperature of the System-on-Chip (SoC). For this reason, we also tried to test DDR2 SDRAMs in a high temperature condition. When the temperature goes up, the more bits flip and the remanence time is much shorter. In other words, at high temperatures, the degradation process is accelerated and is very soon. On the other hand, the cooler the memory, the less bits flip and memory can retain its data for a longer period of time. Therefore the rate of RAM decay is largely a function of temperature. This poses a serious threat to hardware which operates with secret information (typically security key) in a secure environment. We obtained 5 measurements at some delay time in the Fluctuation Region. We measured the DRAM package temperature after the first measurement and it was 28 °C. However, the temperature of the package after the 5th measurement was 55 °C. Note that the package surface temperature is not the actual temperature at the die - which will be much hotter. Based on the experiments, the percentage of flip bits at the 5th measurement was much higher than the first one. For example, in one sets of 5 measurement continuously, the first measurement has around 58% flip bits, but the 5th measurement had almost 64% flip bits at the same delay time, since the DRAM is much hotter at 5th measurement than the first measurement.

3.1.4 Potential Attacks

Remanence characterization results can be used to build secure memories that are less vulnerable to various attacks. Cold boot attack is such an example that makes use of the property that the remanence effect is prolonged by cooling down RAM chips. Cold boot attacks rely on DRAM remanence that allow keys to be recovered from memory even after a machine has been powered down. These attacks can retrieve unencrypted data from RAM; The cold boot attack requires physical access to the machine, a boot disk (or specialized hardware), and quick timing (as data remains resident in RAM for only a short time after its powered down). The extent and predictability of memory remanence and report that remanence times can be increased dramatically with simple cooling techniques [31].

3.1.5 Data Analysis and Evaluation

Here, we present the method of extracting random bit sequences from our TRNG, the uniqueness of the data generated from the designed TRNG, and finally the results of NIST statistical tests for the randomness of the data.

Producing Random Number Generation

Given the data we collected, the next step is to extract random numbers from the data. Our initial attempt was to select an arbitrary delay time during the fluctu-

ation region, and use the memory that was read at that time as our set of random data. However, our analysis found that this data did not reliably generate suitably random data. The data was heavily biased with approximately 60% of bits being 0. As a secondary approach, we examined the differences between fluctuation and startup regions. Simply taking an XOR of the data returns the differential bits from the two regions. In the fluctuation region, we had 168 measurements and these were XOR'ed with a measurement taken from the startup region. Note that the startup region measurements were stable, so we only compared with one measurement from the startup region. Figure 3.4 shows the distribution of the 168 measurements taken from the Fluctuation Region XOR'ed with the measurements collected from the Startup Region. The mean of percentage of 1s among the 168 measurements is 0.5090 which is close to 0.5, the ideal case. The standard deviation of the data is 0.0201 and because it is close to zero, indicates that the data points tend to be very close to the mean of the sets of our measurements. As shown, the shape of the histogram clearly illustrates that the distribution of 1s and 0s is close to normal distribution. Therefore the distribution indicates that DRAM clearly demonstrates the characteristics of a TRNG as it relates to randomness. We were also interested in the distribution of XOR'ed data relative to the delay time (Figure 3.5). We can see that the data is centered around 50% as expected. Moreover, other than the first 200 ms, the variation is not time de-

pendent either, meaning that one could select any measurement in the fluctuation region to use as our random data generator.

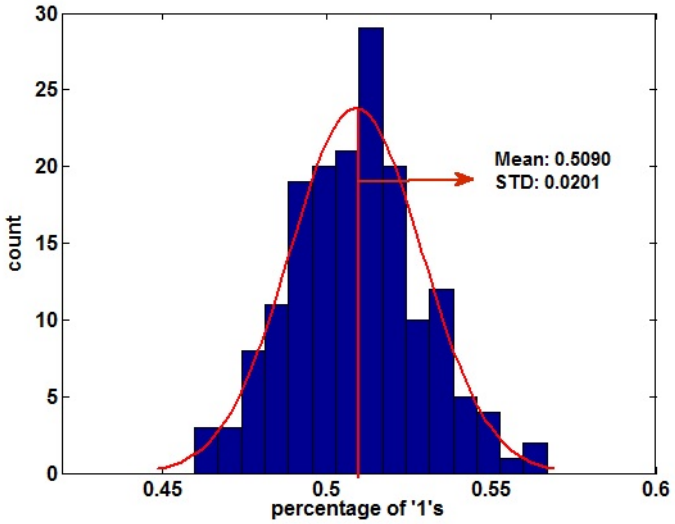


Fig. 3.4: The histogram of the XORed data between Fluctuation and Startup regions.

Investigating Uniqueness of the Data

Since we have a finite set of data, when we regenerate the data, we might get the same set of bits again and that is a security vulnerability because if someone was able to break into the nonvolatile storage where the random bits are stored, they know the sequence of random bits even if we regenerate the bits. For these reasons, we did some experiments where we could generate two different sets of random bits from two different data points from the fluctuation region. Then

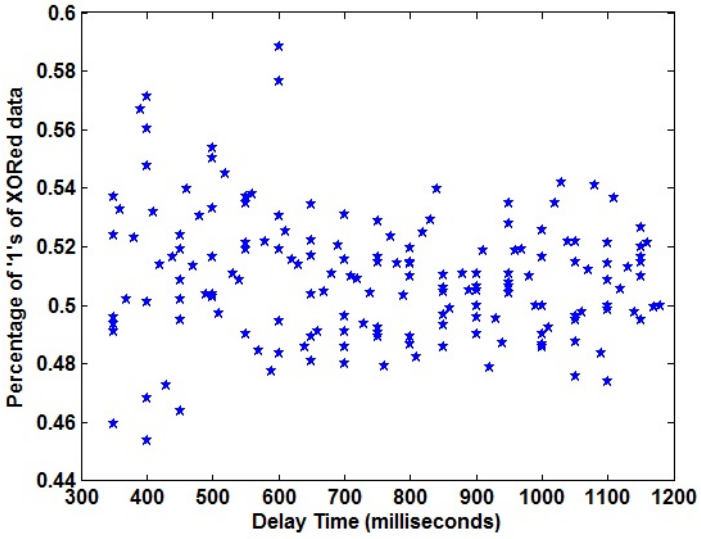


Fig. 3.5: Scatter plot of percentage of '1's of XOR'ed data that are taken from Fluctuation and Startup regions.

compared these sets of random bits to see how different they are. Based on our results shown in Figure 3.6, we can always get new sets of random bits. The mean of the value and the standard deviation are 0.4969 and 0.0344, respectively.

NIST Statistical Test Suite Results

The NIST statistical test suite is used to evaluate the "randomness" of the bit strings produced by DRAM cells. Based on this test, we can determine whether a data set has a recognizable pattern or the process that has been generated is significantly random. The NIST Test Suite (NTS) is a statistical package consisting of different types of tests to evaluate the randomness of binary sequences. Each

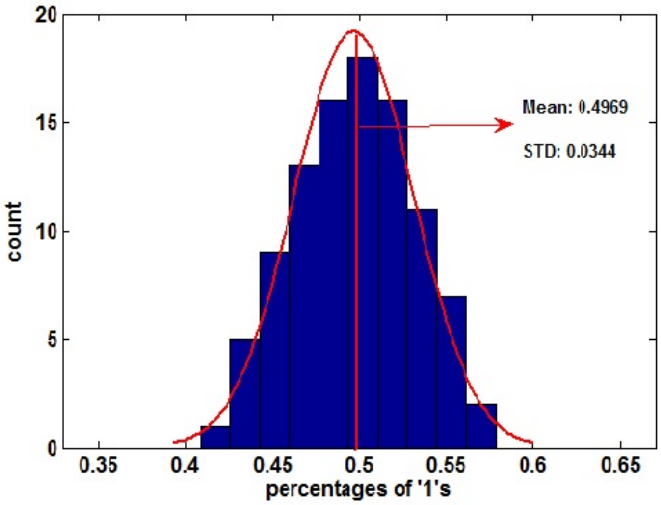


Fig. 3.6: The histogram of the Uniqueness of the data at Fluctuation Region.

statistical test is employed to calculate a P-value that shows the randomness of the given sequences based on that test. If a P-value for a test is determined to be equal to 1, then the sequence appears to have perfect randomness. A p-value $>= 0.01$ (normally 1%) would means that sequence would be considered to be random with a confidence of 99% [87]. To evaluate our collected data from DRAM, we apply NIST tests to the differential bits as discussed in subsection 3.1.5. Table 3.1 displays the average results of a suite of 15 performed NIST tests at room temperature. Table 3.1 shows that all the NIST tests p-value are greater than 0.01, this indicates that the measurements pass the requirements for randomness. We also applied NIST tests to the data gathered at high temperature. For space reasons, we do not show the NIST results, but even at high temperature con-

Table 3.1: NIST Statistical Tests Results at Room Temperature Condition.

NIST Tests	P-value	Result
Frequency	0.6247	passed
Block Frequency	0.6734	passed
Runs	0.7731	passed
Longest Run	0.6457	passed
Cumulative Sums	0.7035	passed
Rank	0.8241	passed
FFT	0.4872	passed
Linear Complexity	0.6576	passed
Overlapping Template	0.5594	passed
Non Overlapping Template	0.9184	passed
Approximate Entropy	0.6943	passed
Universal Statistical Test	0.4208	passed
Random Excursions Variant	0.5557	passed
Random Excursions	0.2843	passed
Serial	0.5208	passed

dition, the data streams passed the NIST tests and high temperature does not have effect on the randomness of the data even though it has a huge impact on the remanence time and decay rate. Note that we were able to get 420 Kbit of data from 1 Mbit of DRAM cells (XOR'ed of data from Fluctuation Region and Startup Region). Therefore, extrapolating to a system with 8 GB of memory, we could generate approximately 3 GB of random data.

3.2 DRAM-based Random Number Generation using its Startup Value Behavior

3.2.1 DRAM Startup Value Behavior

In Chapter 2, we demonstrated that DRAMs surprisingly have startup values - i.e. non-zero values when the DRAM is powered on. While older DRAMs may show potential as a PUF, our work with modern DDR DRAMs show that they not satisfy criteria to serve as a PUF. Specifically, these startup value patterns were neither random or reliable. However, we will show that DDR DRAMs can still be used to generate random numbers. Using a variety of correction mechanisms, we are able to improve the randomness of the numbers such that they pass the NIST tests. Further work will investigate the suitability of the approach on a variety of DRAM parts.

3.2.2 Experimental Setup

Our experiment involved using the on-board MIRA P3R1GE3EGF G8E DDR2 on the Diligent Atlys board. (Xilinx Spartan 6 FPGA). The FPGA hardware was configured to transfer the memory data through a serial port at startup. A serial terminal was then used to record the bits and write them to a text file.

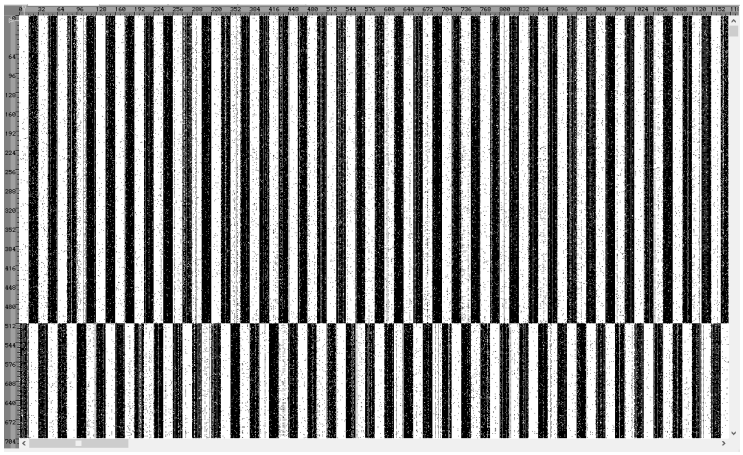


Fig. 3.7: Small section of the bitmap from DRAM for trial 3.

3.2.3 Data Analysis and Results

The use of newer DRAMs for PUFs has tremendous promise because of the large memory space, but it also contains large potential drawbacks. In our experiments, we found the startup values show a clear bias that reflects the architecture of the DRAM. In addition, not all trials produce adequate results. At times, the startup values are completely non-random yielding no valid data to be used for a PUF. More research is needed to see what is exactly causing the DRAM to start-up to these modes and if it can be avoided. Thus multiple trials will be needed to ensure that the bits are behaving correctly upon startup. We show a graphical representation of a sample bitmap from various trials in Figures 3.7, and 3.8. Each row in the Figure represents 8192 bits where white is 0 and black is 1. Across the whole device, clear patterns could be seen when mapping the data to a bit map.

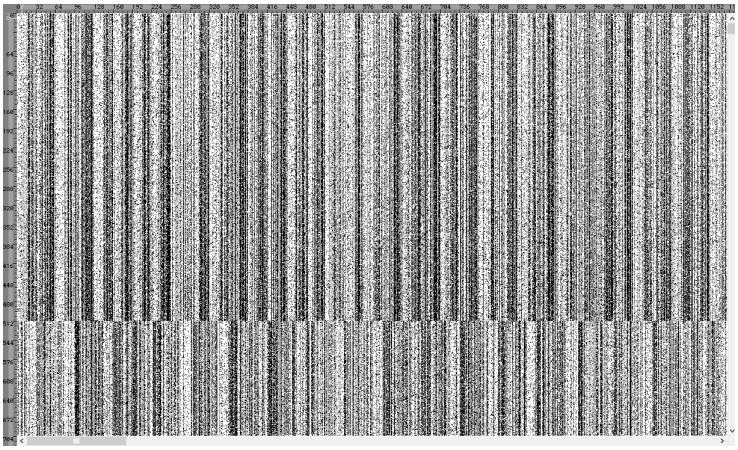


Fig. 3.8: Small section of the bitmap from DRAM for trial 6.

The architecture of the DRAM is what is likely strongly influences this style of DRAM. These patterns can be observed across multiple trials. This particular DRAM alternate between ones and zeros every 16 bits consistent with the DRAM 16-bit width. Every four megabits, the DRAM has a section of 32 bits before returning to its pattern. Figure 3.7 shows a small subsection of the bitmap of one of the trials on the DRAM. The pattern described above is heavily noticeable in the figure. However, not every trial gives these results. Some of the trials had far less stable bits and the patterns were less obvious. In Figure 3.8, you can see the a much larger percentage of bits that don't follow the pattern compared to Figure 3.7. In Figure 3.8 despite the higher variance it bits, the original pattern is still easily discernible to the human eye despite the significantly larger number of bits that do not follow the pattern. Of the six trials, Figure 3.7 appeared to follow the pattern the best while the trial from Figure 3.8 followed the pattern

the least. The other trials were anywhere in-between. Another interesting result comes when one looks at how the pattern starts. In Figures 3.7 and 3.8, one can observe that the pattern starts with 16 bits of zeros first before alternating to 16 bits of ones. However this is not always the case.

In other trials, the pattern starts off with 16 bits of ones instead of zeros! So not only is each trial unique and unpredictable with which and how many bits will be affected, a completely inverted pattern can also be achieved. Clearly, the use of DDR DRAM as a PUF is unlikely. With 1 Gbit of memory read, there appears to be no sections on the DRAM that produce both unique and stable bits to be used to generate challenge response pairs. This unpredictability helps us however. Which bits and how many bits that remain stable were extremely unpredictable from trial to trial. Therefore it would be possible to xor multiple trials to introduce the random bits from multiple trials. By xoring multiple trials, we can combine the unpredictability in the bits that do not follow the set patterns. Thus introducing more trials to xor would likely make the results better. It is unlikely that the entire or even a majority of the DRAM can yields random data by xoring multiple trials. However several sections likely will pass randomness tests. The more trials that are included, the more sections will pass. However when xoring 4 different trials together, we get that 1.29% of the blocks pass all the tests. With a 1 Gbit DRAM, this gives us approximately 1.65 MB of random

bits that we can use for a secure random code.

NIST Test

As seen in Table 3.2, xoring the trials gives significantly higher passing rates for some of the later tests that were producing insufficient number of passing blocks. More importantly when you look at the percent of passing all tests, we finally achieve blocks that pass every test.

Van Neumann Corrector

Another approach is to apply the Von Neumann Corrector [55] on all the pairs of bits. This works by discarding all 00 and 11 pairs and correcting a 10 to a 1 and a 01 to a 0. Since such a large part of the data has 16 bits of ones or zeros in a row, we can essentially remove such sections and only look at the unstable bits. After applying the corrector, we again used blocks of 65536 bits, and we tested the maximum number of blocks allowed by the new block size. The remaining bits that did not fit into a full 65536-bit block were ignored. Discarding these bits reduced the number of random bits available. Table 3.2 shows the result of using the Von Neumann approach. As expected, the amount of data that is available to select a random key is greatly reduced. Of the six trials, the largest key space is 35.1MB or 27.4% of the original DRAM and with the smallest at 9MB or 7.0% of the original DRAM. However, the approach shows some promise

Table 3.2: DRAM trials and corresponding NIST tests

NIST Test	Trial 1	Trial 2	Trial 3	Trial 4
Frequency	4.99%	7.48%	31.97%	1.13%
BlockFrequency	19.20%	79.84%	95.61%	19.72%
CumulativeSums	4.98%	7.87%	33.29%	1.12%
CumulativeSums	5.28%	8.63%	33.31%	1.23%
Runs	2.35%	0.00%	0.00%	0.09%
LongestRun	16.91%	0.00%	0.00%	5.54%
FFT	0.01%	0.00%	0.00%	0.00%
ApproximateEntropy	0.02%	0.00%	0.00%	0.00%
Serial	27.13%	0.00%	0.00%	0.01%
Serial	95.11%	0.00%	0.00%	5.93%
Pass all tests	0.00%	0.00%	0.00%	0.00%
	Trial 5	Trial 6	Trial 1,3 xor	Trial 1,4,5,6 xor
Frequency	9.20%	0.20%	0.00%	13.12%
BlockFrequency	35.29%	88.92%	0.02%	19.55%
CumulativeSums1	9.42%	0.20%	0.00%	13.19%
CumulativeSums2	9.71%	0.21%	0.00%	13.24%
Runs	0.00%	0.13%	0.00%	18.84%
LongestRun	0.19%	0.18%	0.78%	27.38%
FFT	0.00%	0.41%	1.96%	75.42%
ApproximateEntropy	0.00%	0.00%	0.14%	26.14%
Serial1	0.00%	0.00%	29.95%	64.87%
Serial2	0.02%	0.07%	91.88%	95.41%
Pass all tests	0.00%	0.00%	0.00%	1.29%

with respect to randomness in that it does significantly better than the raw trial results. Unfortunately, it fails the vast majority of the runs tests. Either the bits are changing alternating between runs of ones and zeros too often or not often enough. Since the pattern switches every 16 bits, trials with a large number of unstable bits might alternate too slowly because if it has too many outliers in a 16 bit section. This is supported by looking at the data size of the Von Neumann corrector output.

The two smallest datasets (Trials 2 and 3) had the best performance on the limiting test, the runs test. This is important to note because the Von Neumann corrector is designed to remove sections of the data that do not have significant number of outliers. Thus, it seems that with less outliers they are more random. Perhaps the more unstable bits there are, the more predictable they become. We come to this conclusion because without any unstable bits, the entire data set would be removed. Thus, the more unstable bits, the larger the trials will be after applying the Von Neumann corrector. Because the smaller trials performed better, its possible that as more unstable bit show up, they are heavily biased in locations leading to a less random result. A greater number of outliers or unstable bits leading to less random data after the Von Neumann correction is somewhat unexpected. However, this interesting paradox can be solved by combining multiple trials by xoring them. If sections of the unstable bits become predictable, it

Table 3.3: DRAM trials with Von Neumann corrector

	Trial 1	Trial 2	Trial 3
Data size	35.1MB	12.8MB	9.0MB
Frequency	97.09%	75.91%	71.97%
BlockFrequency	99.98%	99.39%	98.08%
CumulativeSums	97.68%	78.05%	74.11%
CumulativeSums	97.74%	77.80%	74.05%
Runs	0.11%	1.94%	12.31%
LongestRun	37.06%	60.08%	77.95%
FFT	96.48%	98.16%	98.54%
ApproximateEntropy	19.46%	51.07%	72.22%
Serial	95.91%	97.20%	98.22%
Serial	98.64%	98.78%	98.86%
Pass all tests	0.0350%	0.6248%	5.5974%
	Trial 4	Trial 5	Trial 6
Data size	28.3MB	24.0MB	34.6MB
Frequency	93.46%	89.85%	96.21%
BlockFrequency	100%	99.92%	99.99%
CumulativeSums	94.62%	91.50%	97.05%
CumulativeSums	94.66%	91.48%	96.95%
Runs	0%	0.02%	.02%
LongestRun	24.22%	17.76%	29.13%
FFT	96.40%	96.24%	95.47%
ApproximateEntropy	16.81%	11.14%	11.39%
Serial	27.13%	94.95%	94.90%
Serial	95.11%	98.62%	98.58%
Pass all tests	0.00%	0.0085%	0.00%

will be canceled out by the unstable bits from other trials. Looking at results of xoring trials and then performing Von Neumann corrector (Table 3.4), we see a much higher pass rate. The resulting data were all very similar in size with all of them falling in the 25-27% of the original file. All trials had at least 56% of trials pass every test. This is a huge improvement upon earlier results. The best trial for just using the corrector resulted in 5.6% of trails passing and xored trials best resulted in 1.29%.

Combining Two-methods

Combining these two methods clearly can be used to generate random keys from the unstable bits. Our results can generate at a minimum 15.12% and a maximum 20.45% of usable random bits. With 1 Gbit of memory to choose from, 15-20% is a sufficiently large enough to generate a secure random key. Also with xoring the trials, we had to xor 4 different trials before getting a small amount of bitstreams to pass. Although xoring more trials produced better results, with this method we saw sufficient results from just xoring two different trials. As demonstrated earlier, the DRAMs have patterns in there startup behavior. Moreover, these patterns are not consistent, as they differ greatly from trial to trial. It is possible that the DRAM has a certain number of startup states or modes that it can enter. Comparing the different trials, two of them had 94.14% of the bits in common. It is

Table 3.4: DRAM trials XOR with Von Neumann corrector

	Trial 1,2 xor	Trial 1,3 xor	Trial 1,2,4,5 xor
File size	34.4MB	33.3MB	33.9MB
Frequency	94.71%	94.73%	95.61%
BlockFrequency	99.43%	99.32%	99.11%
CumulativeSums	95.33%	95.34%	96.08%
CumulativeSums	95.31%	95.21%	96.11%
Runs	80.67%	85.13%	93.80%
LongestRun	94.34%	96.20%	97.70%
FFT	98.53%	98.69%	98.77%
ApproximateEntropy	80.87%	85.70%	89.95%
Serial	98.49%	98.55%	98.70%
Serial	98.94%	99.00%	98.90%
Pass all tests	56.29%	62.61%	74.81%
	Trial 1,3,4,6 xor	Trial 1,4,5,6 xor	Trial 1,2,3,4,5,6 xor
File size	33.1MB	33.0MB	33.6MB
Frequency	93.42%	93.80%	95.02%
BlockFrequency	99.42%	97.35%	98.51%
CumulativeSums	94.25%	94.46%	95.55%
CumulativeSums	94.19%	94.34%	95.41%
Runs	82.90%	95.82%	97.80%
LongestRun	95.40%	98.26%	98.66%
FFT	98.67%	98.90%	98.74%
ApproximateEntropy	85.49%	91.13%	91.92%
Serial	98.64%	98.76%	98.88%
Serial	98.98%	99.06%	98.96%
Pass all tests	59.70%	73.40%	77.92%

entirely possible that the remaining ≈6% is just unstable bits from this particular startup mode.

Supporting the theory on startup modes, the 16-bit pattern sometimes starts bits set to one and sometimes the bits set to zero. This means a section of mostly stable bits can be all ones one trial and all zeros the next, making it highly likely that some sort of modes exist. More testing is needed to see if reason for the inconsistent numbers and distribution of unstable bits can be determined. The DRAM having a startup mode: would not completely hinder its application. It would reduce the number of challenges, but an attacker would have no way to know which mode it is using and would be unable to recreate the seed state. If the mode can be determined the DRAM would be less secure for generating random numbers. This would allow an attacker with physical access to replicate the seed state. Furthermore all of this behavior is untested and unconfirmed in other DRAMs of the same type. More testing is needed to determine if the startup values are based on modes and if other DRAM have modes and if so do the modes behave the same as the DRAM we tested. Despite these challenges, the DRAM might still have application in cybersecurity.

3.3 Potential Attacks

Tests will need to be preformed on more devices to see if they a perform similarly. If all DDR DRAM behave in this way, the only way for an attacker to reproduce a key would be with physical access to the device that was used to generate the key. This ability to reproduce is also limited with the varying amount of amounts of unstable bits in each trial. Further if we choose to xor the trials and apply the Von Neumann corrector, it would become even harder to for an adversary to reproduce the results without knowing the details of each trial that was xor'ed. It is possible for an attacker with knowledge of the memory architecture to probe a DRAM and learn the factors that cause the varying amount of unstable bits. However, this would have little benefit to the attacker, as there is no standard for error in a key. An attacker correctly predicting 90% of the bits won't help as one wrong bit is all that is needed to deny the attacker access.

3.4 Conclusion

In this chapter, We propose a new method for generating hardware true random number generators and studied the potential of using remanence effect of the DRAM cells for constructing a high quality TRNG. Our new constructed TRNG has all the characteristics of a strong TRNG and the approach to build this new TRNG is completely novel and concise. The results from NIST Tests

clearly shows that the remanence based DRAM TRNG is a promising candidate for cryptographic systems. We also investigated the high temperature effect on data remanence and randomness of the data. Our results shows that even at high temperature, the measurements pass the requirements for randomness. We also demonstrated the use of DRAM startup values to produce a TRNG. It has been shown that DRAM startup values have potential as a physical unclonable function (PUF). However, in our tests using a newer DDR2 DRAM, we demonstrate that the startup values are not suitable for device authentication but they do have use for creating random keys. We have developed an approach to extract random numbers from the DRAM startup values. Finally, we assessed and tested the randomness of selected data by applying the NIST Statistical Test.

Chapter 4

A Study of Power Supply Variation as a Source of Random Noise

4.1 Introduction

In this chapter, we consider the construction of True Random Numbers Generators (TRNGs) using variations in power supplies. We demonstrate that power supply line outputs do not have a constant voltage and the variations in voltage follow a normal distribution. These variations can be used to create truly random bits that demonstrates a high entropy rate based on the results obtained from the NIST Statistical Test Suite. In order to quantify the impact of variations on the input signal of a circuit, we analyze the impact of such variations using Monte- Carlo simulations as well as an actual implementation. Results were obtained for evaluating the accuracy and randomness of the data gathered from our proposed circuit. A detailed analysis of the effect of variations of different power supplies is also presented with observations on their usefulness as a TRNG.

The key advantage of our power supply variation based TRNG is its simplicity of implementation.

4.2 Challenges, Motivations, and Objectives

In this Chapter, the effects of voltage variations inherent to power supplies are examined. We focused more on studying the impact of power supply noise on the behavior of VLSI circuits (inverters chain). Our TRNG model is increasingly sensitive to supply voltage noise. Investigation of the variation on power supplies for constructing TRNGs yielded some unique advantages: 1). First, our proposed TRNG model purely relies on the input voltage variation that is coming from the power supply. However, other TRNG models, such as the Ring Oscillator (RO) based TRNG, are not only influenced power supply variation, but also affected by a variety of other properties, so it is hard to say what exactly is causing variations in the RO TRNG model. In fact, a lot of variations are based on timing variations. 2). Our model is a very simple circuit to build, so it does not add significant overhead (cost, voltage, etc,) to the system's normal operation. 3). Our simple circuit can continuously generate bits unlike a memory based TRNG where a fixed number of bit strings are generated.

Challenges: In the work by Stipčević and Koç‘ [92] they mentioned that the majority of TRNGs models, whose speed is at least 1 Mbit/sec, are too expensive

to implement and far too bulky to make practical. As the need for practical TRNG solutions grows with our utilization of TRNGs in everyday devices and practices, there is a desire to minimize the cost required for implementation of these models. These concerns culminate as a challenge to produce a low-cost and practical design implementation for noise-based TRNGs.

Motivations: General TRNGs are constrained by two major issues: (1) that they are too expensive to implement and (2) they tend to be bulky and not efficient. Our work focuses on presenting a low-cost TRNG solution, in terms of overheard and design expenditures that can provide a large number of random bits and is very easy and simple to implement. A potential boon of centralizing the TRNG around power-supply noise is that we could create a model that works as a constant generator for bit strings.

Objectives: The primary objective of our work is to create a simple true RNG based on power source noise and variation that has the least cost, in terms of area overhead and additional electronic circuit elements, and provides good randomness. In order to achieve our goal, we proposed a dynamic voltage feedback tuning system that allows for practical implementation. By adding the feedback loop we provide stability to the system while removing human error from the process, creating a robust and self-correcting design.

4.3 Power Supply Noise under Study

The variations in power supplies have risen to become one of the most important challenges and a higher level of concern as technology has scaled down in new digital designs. Due to the trends of higher switching speeds, more active pins per package, a decrease in supply voltage required for ICs, and more finely tuned circuit characteristics there is a much larger ratio of the peak noise voltage to the ideal supply voltage. Nominal supply voltage is gradually scaling down with newer technologies, but threshold voltage scaling has all but stopped. Consequently, circuit delay sensitivity to the margins is increasing with each technology node. Power supply noise is primarily a voltage drop in power distribution networks resulting in different voltages at different parts of the same chip. Two of the reasons contributing to these behavior variations are the resistive and inductive voltage drops across power supply networks. An example of this behavior would be the activity of any switches in the device circuitry. As the switch opens and closes, this action causes power be pulled or driven which results in a fluctuation of the power causing noise.

4.4 Dynamic Variation in Power Supplies vs Voltage Regulators

To determine a Gaussian distribution, which is a factor of proving the non-cyclostationary behavior of the power supplies, we take a look at the math and

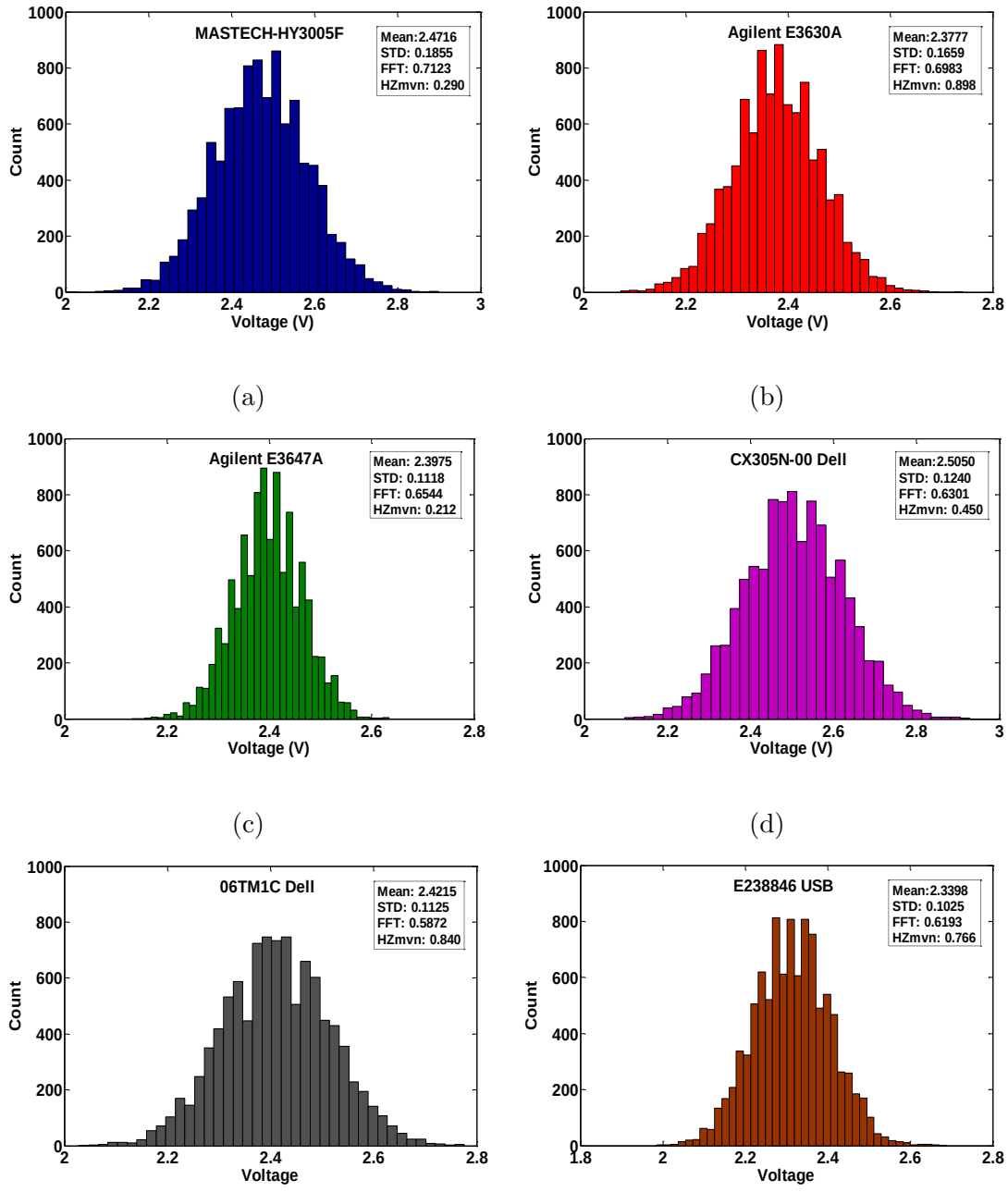


Fig. 4.1: Histograms of a portion of the data (10000 samples) directly from six different power sources (Bench power supply, USB, Computer power source, DC power supply).

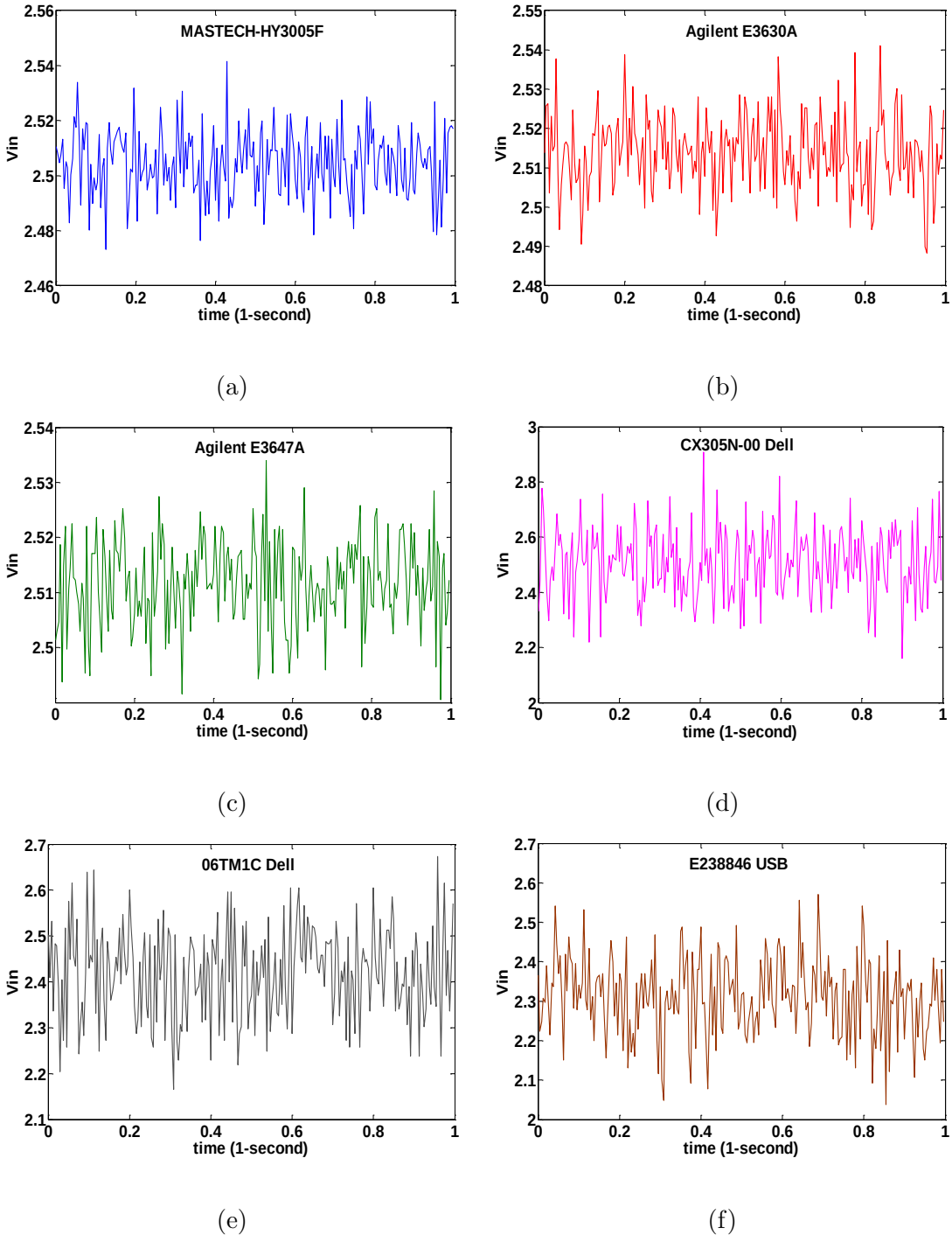
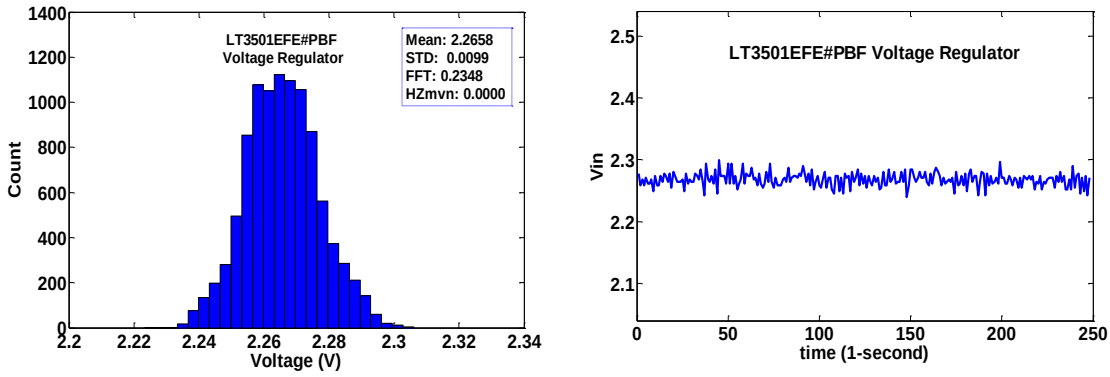


Fig. 4.2: Input voltage plot of 1-second (240 samples) of the data from 6 different power supplies.



(a) (b)

Fig. 4.3: Using voltage regulator (LT3501EFE#PBF) for plotting: (a) Histogram of data (10000) samples, (b) Input voltage of 1-second (240 samples) of the data.

theory behind this definition. A Gaussian distribution [$X \sim \mathcal{N}(\mu, \sigma^2)$] in a variant X with mean μ (or expectation of the distribution) and variance σ^2 (σ is standard deviation) is a statistical distribution with probability density function:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{4.1}$$

In order to prove that the noise from power supplies are not periodic and cyclostationary, we examined six different power sources which are listed in Table 4.1. For each power source, we took 10,000 samples (at 240 samples/second) and their distributions are shown in Figure 4.1. The reason for the 240 samples/second is that we read data at a baud rate of 9600 (i.e. 9600 bits/second). Since each byte of data requires 10bits to send and we required 4 characters per

data sample the rate is effectively 240 samples/second. The data collected for Figure 4.1 was done using an Arduino to capture signals directly from each power supply. Previous work by Burleson and Mirza [13] claimed that signals from a power supply are cyclostationary. In our analysis of power signal data, however, we found that the sample distribution is in-fact non-cyclostationary and does not cyclically vary in time. As can be seen in the figures, the Mean, Standard deviation (STD), Fast Fourier Transform (FFT), and HZmvn of the sample sets have been calculated. Our FFT metric from the NIST test [87] gives values that range from 0.5872 to 0.7123 showing that there there are no determinable cyclic patterns. Note that Henze-Zirkler's Multivariate Normality (HZmvn) test is based on a non-negative functional distance that measures the distance between two distribution functions: the characteristic function of the multivariate normality and the empirical characteristic function. This test can confirm that analyzed data has a normal distribution if the p-value associated with the Henze-Zirkler statistic is greater than 0.05 [39]. Based on the metrics that have been calculated for these distributions, it is clear that the power sources demonstrate a standard Gaussian process. Examining Figure 4.1 one can easily see that the calculated values for the power sources all fall within acceptable range. Figure 4.2 shows the time domain plot of the data gathered from the investigated power supplies. The x-axis and y-axis show 1-second of the time and output voltage, respectively. As

can be seen in this figure, the power supplies are not showing a constant 2.5V but also have continuous variation in their voltage waveforms. Note that we did not make the initial assumption that the power supplies were ideal and instead begin with the assumption that the power supplies create a deterministic output. It was only after experimental measurements, which included the effect of internal resistance of the power supplies, that we found that the output produced a stochastic component.

Power Source	Output Voltage
MASTECH HY3005F Bench Power Supply	2.5V
Agilent E3630A Bench Power Supply	2.5V
Agilent E3647A Bench Power Supply	2.5V
Dell CX305N-00 Computer Power Supply	2.5V
Dell 06TM1C DC Laptop Adapter	19.5V
E238846 USB Cable attached to Dell PC	5V

Table 4.1: Power sources under evaluation

Similar to the experiments we did for proving non-cyclostationary behavior of the power sources, we repeated the same procedure with a voltage regulator (LT3501EFE#PBF). As shown in Figure 4.3 (a), the calculations for STD, FFT, and HZmvn are much lower than those of the other six power supplies examined. The most glaring issue is that the HZmvn value is 0.0000, which is far below the 0.05 threshold required. Furthermore, the spread of data is very narrow which

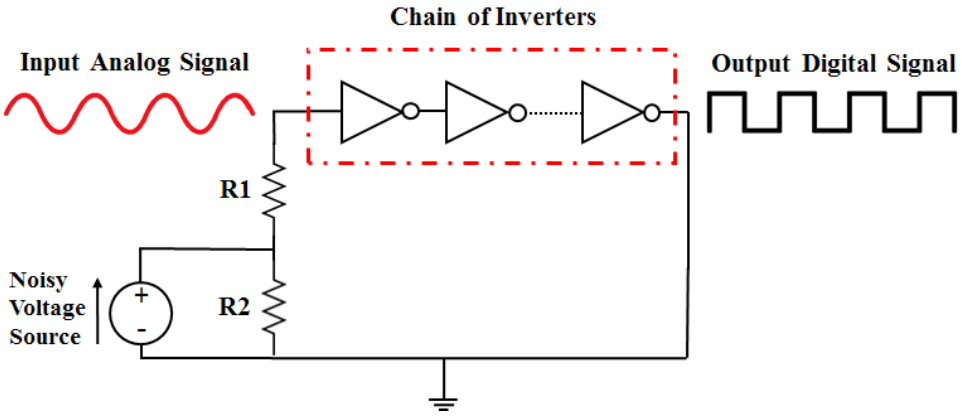
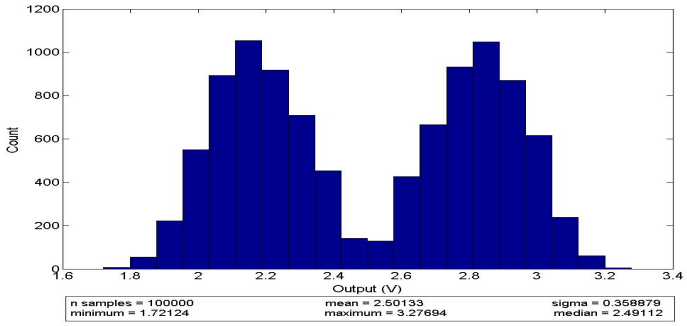
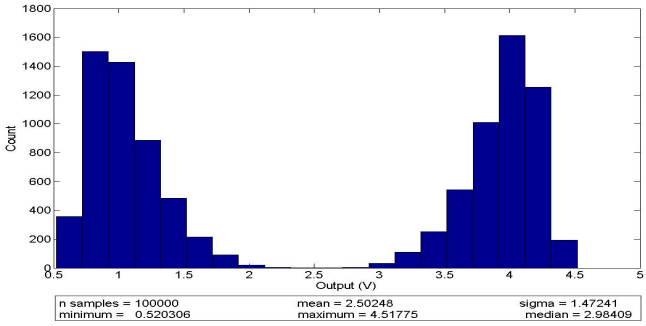


Fig. 4.4: A schematic of our TRNG circuit design.

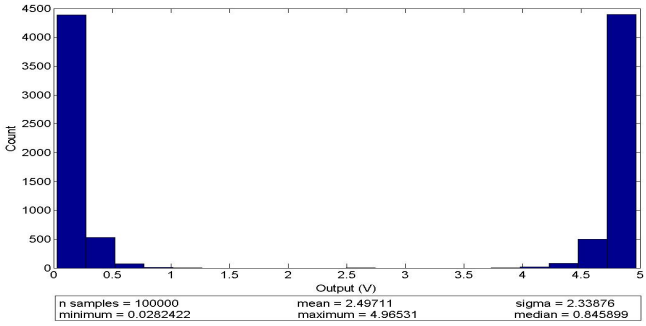
indicates the STD value is very small. Figure 4.3 (b) shows input voltage plot of 1-second (240 samples) of the data from the voltage regulator which has a very small variation of less that 0.02 at most. Based on these figures (4.3 (a) & 4.3 (b)) we can conclude that the data from the voltage regulator does not have enough variation to be used for random number generation. While the results of the voltage regulator in Figure 4.3 (a) appears to match the Gaussian properties desired, deeper investigation of Figure 4.3 (b) reveals that there is not enough variance to be suitable for generating TRNG. Without sufficient variance, it becomes difficult to extract noise differentials in order to produce random bits. The behavior of the voltage regulator is not surprising, since the purpose of the voltage regulator is to provide a stable voltage output.



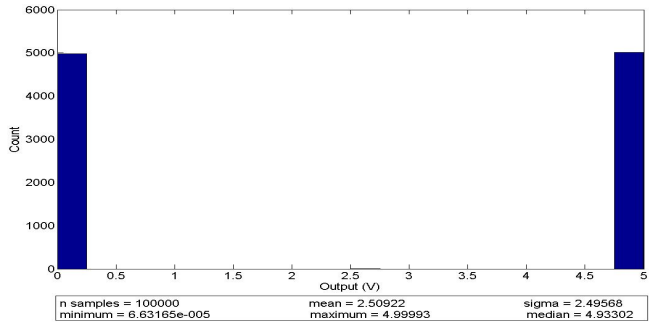
(a)



(b)



(c)



(d)

Fig. 4.5: Plots of output voltage vs the number of samples while using (a) 1-inv, (b) 2-inv, (c)3-inv, and (d) 4-inv in chain using Monte Carlo simulation.

4.5 Preliminary Power Supply-based TRNG Circuit Model

The noise generated by the various power supplies shows promise based on our measurements shown in the previous section. However, in order to construct a TRNG, we needed to design a circuit capable of taking as input the signal coming from a power supply and generating a series of random bits. While the noise from power sources is non-periodic and non-deterministic, the circuit element is required to convert the Gaussian distribution into uniform distributed sequences of 0's and 1's. Since the Gaussian distribution is centered at the mean, i.e. half of the distribution is greater than the mean and half is less than the mean, our circuit only need discriminate between voltages above and below the mean; to produce strings of 0's and 1's. In other words, this circuit element is responsible for the conversion of analog behavior to digital behavior based on a predetermined threshold. Passing the signal through an inverter should provide the necessary discrimination assuming that the power supply mean is equal to the transition point of the inverter. For a balanced inverter, this point is $\frac{V_{DD}}{2}$. However, since the input is very close to $\frac{V_{DD}}{2}$, the output of an inverter will still be very close to $\frac{V_{DD}}{2}$ as well. To counteract this, instead of a single inverter, we propose using a chain of inverters. These inverters act as amplifiers to amplify the input variations and push the output away from $\frac{V_{DD}}{2}$ to 0 and V_{DD}. By using the chain of inverters, we are able to amplify the discrimination spread of voltage readings. Figure 4.4

shows a schematic of our primary TRNG circuit design. The purpose of the chain of inverters is to transform the analog signal into a digital form. As indicated earlier, the role of the inverters is to amplify the input signal and to "convert" its value into a binary representation; to 1's & 0's. Based on our observation after implementing the circuit, we found that the greater the number of inverters in the inverter chain, the better the accuracy we get out of the circuit. "Accuracy" is defined as the displacement between the two Gaussian peaks for the purpose of having voltages appear in one of two extreme ranges (e.g. 0V, 5V). Accuracy is measured as a percentage of all recorded outputs that meet the ideal case. The voltage divider is kept for the non-bench power supplies in order to bring the voltage close to 2.5V.

4.5.1 Simulation Results

Figure 4.5 clearly demonstrates the effects of an increasing number of inverters on the percentages of the accuracy of the output. In this figure, the x-axis represents the voltage levels from 0-5V and the y-axis shows the number of readings that registered that voltage level. Figure 4.5 (a), (b), (c), and (d) show the simulation result, while considering the design with one, two, three, and four inverters using OrCAD Capture's Monte Carlo simulation. One should note that x-axis in Figure 4.5 (a) is at a shorter range compare to (b), (c) and (d) in order to

better illustrate the accuracy of the output. In all, by increasing the number of inverters in chain the mean value will come close to the ideal case which is 2.5 and standard deviation will become larger. This shows that we can completely distinguish between zeros and ones generated from our proposed model.

4.5.2 Implementation Results

Using the graphs in Figure 4.6, we can visually select the ideal voltage that will provide the desired 50% 1's for the TRNG for each examined power supply. While 50% 1's is a necessary property of a TRNG, it is not a sufficient metric of randomness. We, therefore, use the NIST Statistical Test Suite to assess the randomness in a long sequence of bits [87]. NIST has documented 15 statistical tests, and for each test the χ-square variation (χ^2) of a particular parameter for the given bit sequence is compared with that obtained from the theoretical studies of an identical sequence under the assumption of randomness. The χ^2 data is used to generate a p-value which is the probability that an ideal random number generation would produce a sequence less random than the sequence under test. A p-value $>= 0.01$ would mean that sequence would be considered to be random with a confidence of 99% [87]. Table 4.2 shows the results of applying NIST tests to our data using 6 different power supplies. Each power supply's results came from the 4 and 5 inverters chain measurements. As shown in Table 4.2, our data collected for our

TRNG model passed all 15 tests successfully.

At first glance, our simple TRNG circuit may look similar to that of a Ring Oscillator (RO) TRNG circuit. However they are completely different in terms of source of variation and even the circuit structure. A ring oscillator is a device composed of an odd number of logical inverters whose output oscillates between the two logical levels. The inverters are attached in a chain, where the output of the last inverter is fed back into the first; causing oscillation. Our TRNG circuit architecture only incorporates a power supply as input and a chain of inverters without any form of feedback. In further comparison, the source of variation in an RO TRNG is due to inherent oscillator jitter that causes timing differences and metastability in registered outputs. For our model, however, the variations originate from the power supply and we use several inverters to amplify that variation. The only similarity between these two TRNG implementations is the chain of inverters.

While we were able to demonstrate good results from our primary TRNG circuit design, it still has practical limitations due to the need to ensure that the input voltage is centered around $\frac{V_{DD}}{2}$. To overcome this limitation we add a feedback system to allow for better robustness of the system, as we will show in the following Section.

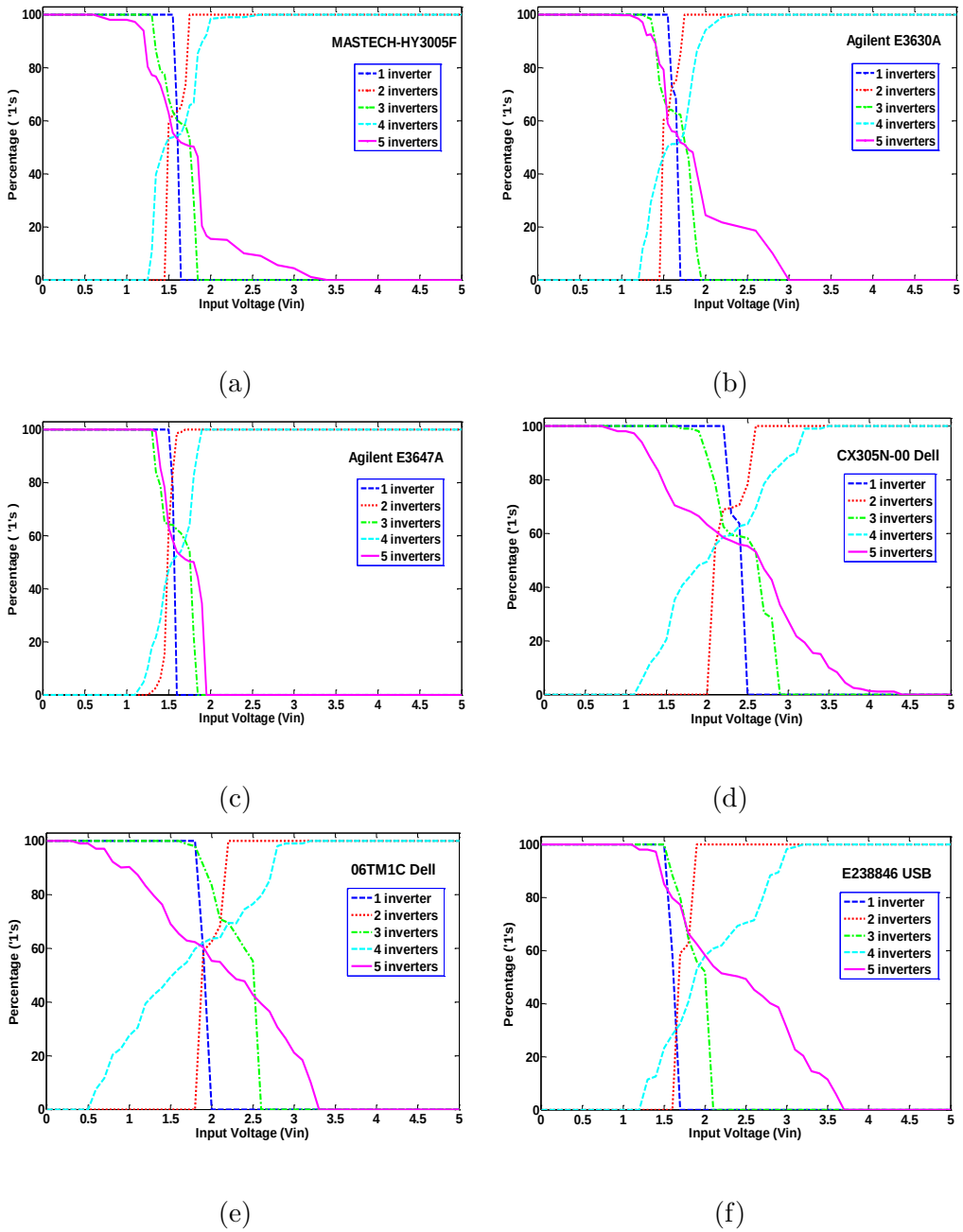


Fig. 4.6: Percentages of 1's at different input voltage level (0-5 volts) after using different number of inverters in inverter chain (1-5 inverters) from 6 different power supplies.

4.5.3 Experimental Setup

The experimental setup essentially consists of four parts: the power source, a chain of inverters comprised of 7404 TTL inverters, an FPGA to sample the data bits and a host PC for collection. Using TTL 7404 series inverters, we considered 1, 2, 3, 4, and 5 inverters for the inverter chain, tested each condition and collected the data from the circuit to analyze the results. The FPGA samples data using a register clocked at 100MHz. Figure 4.6 shows the results we have obtained from using a different number of inverters in the inverter chain. The x-axis indicates the input voltages (0-5 V) supplied to our simple circuit and the y-axis indicates the percentage of value 1s in accordance with a certain input voltage. Figure 4.6 can clearly demonstrate the effect of additional inverters attached to each of the six different power supplies. These figures show that as the number of inverters in the chain increase, there is a greater divergence on the percentage of 0's and 1's collected.

4.6 Advanced Power Supply-based TRNG Model using Tuning System

4.6.1 Tuning System

Normally, electrical circuits are designed to operate with a constant and unchanged range of voltages, making use of the physical property of a circuit in

Table 4.2: NIST Statistical Tests Suite average p-value results of our proposed TRNGs using different types of power supplies.

NIST Tests	p-value **MASTECH HY3005F**	p-value **Agilent E3630A**	p-value **Agilent E3647A**	p-value **CX305N-00 Dell**	p-value **06TM1C Dell**	p-value **USB E238846**
Frequency	0.6454	0.6791	0.7488	0.7293	0.5541	0.6923
Block Frequency	0.4572	0.5583	0.5429	0.8536	0.6402	0.7538
Runs	0.7290	0.5283	0.6203	0.6370	0.4721	0.7199
Longest Run	0.6212	0.6502	0.5826	0.7196	0.5836	0.7044
Cumulative Sums	0.7129	0.8094	0.9360	0.8854	0.8107	0.7547
Rank	0.7721	0.5408	0.7963	0.6628	0.7157	0.3845
FFT	0.8129	0.7553	0.6952	0.8562	0.7735	0.5805
Linear Complexity	0.7334	0.8973	0.8502	0.9163	0.8490	0.8068
Overlapping Template	0.6973	0.5098	0.5663	0.4725	0.6829	0.5812
Non Overlapping Template	0.7420	0.4579	0.5538	0.6390	0.7712	0.5403
Approximate Entropy	0.5641	0.7673	0.9552	0.8854	0.9526	0.8882
Universal Statistical Test	0.4519	0.5236	0.4797	0.6188	0.6482	0.3214
Random Excursions Variant	0.5911	0.3829	0.7122	0.8375	0.5594	0.8604
Random Excursions	0.8523	0.6574	0.6781	0.8953	0.6683	0.6950
Serial	0.6534	0.8836	0.5273	0.9372	0.8705	0.9123

order to produce a desired behavior and function. Fluctuations in the voltages may cause alterations to the operation of the entire electronic system. For this reason, various research has been made into the design and application of voltage regulation for the purpose of maintaining the actual functionality of a specific circuit and the rest of the system. As tailored circuits run in expected environments, the ideal operating point of a given system will change over time, thus the system

will no longer be operating within optimal parameters. To alleviate this problem, one must adjust the system to maintain its performance and functionality in an optimal range (e.g. tuning the system). In other words, the purpose of tuning a system is to customize its behavior in an optimal operation manner. The act of tuning will most definitely add circuit overhead to the PCB, the design, and the development of a given system. The advantage of this is that once the operation of a circuit has been tuned within some threshold, this removes the need for, and errors caused by, human interaction and environmental changes. The adaptive methodology of tuning circuits allows for more robust systems. Pandey et al. [74] examined several works making use of adaptive and self-tuning control (STC) approaches for load-frequency control in power systems, showing their effectiveness in improving system performance. In 2015, research work was done [6] to implement generalized frequency domain controller tuning for voltage sourced converter (VSC) systems, showing that active variations in power can be damped within tolerable levels.

4.6.2 Dynamic Voltage Feedback Tuning (DVFT) Design

To produce a new and more robust design of our primary TRNG circuit, we added a few new elements to allow for self-calibrating design and improved functionality. Our new TRNG circuit design removes the need for manually finding the ideal

voltage and moreover, keeping it steady during operation, allowing for the system to arrive at this voltage automatically using dynamic voltage tuning. Our design solution is a feedback circuit that calibrates the voltage until we have reached the optimal setting, as shown in Figure 4.7. Building upon the previous design, Figure 4.4, our addition to the TRNG circuit is the Dynamic Voltage Feedback Tuning (DVFT) which includes a buffer (B1), a precharged capacitor (C), and a transistor (T1). The buffer, implemented through the use of an inverter, is used to isolate the TRNG output from the feedback circuit. The capacitor serves as a integral controller essentially summing up the past history of 1's and 0's. Finally, the transistor serves as the mechanism to tune the voltage by varying the effective resistance in the voltage divider. The advantage of this self-adjusting system is that it can maintain itself within operation range. In other words, the system is flexible enough to recover from fluctuation in voltage coming from the power supply by attempting to prevent potential mistakes and malfunctions from occurring. As shown in Figure 4.7 it is a very simple design that barely adds overhead to the entire TRNG system.

To better understand the functionality of the proposed circuit, we need to analyze the voltage and current flowing throughout the circuit model.

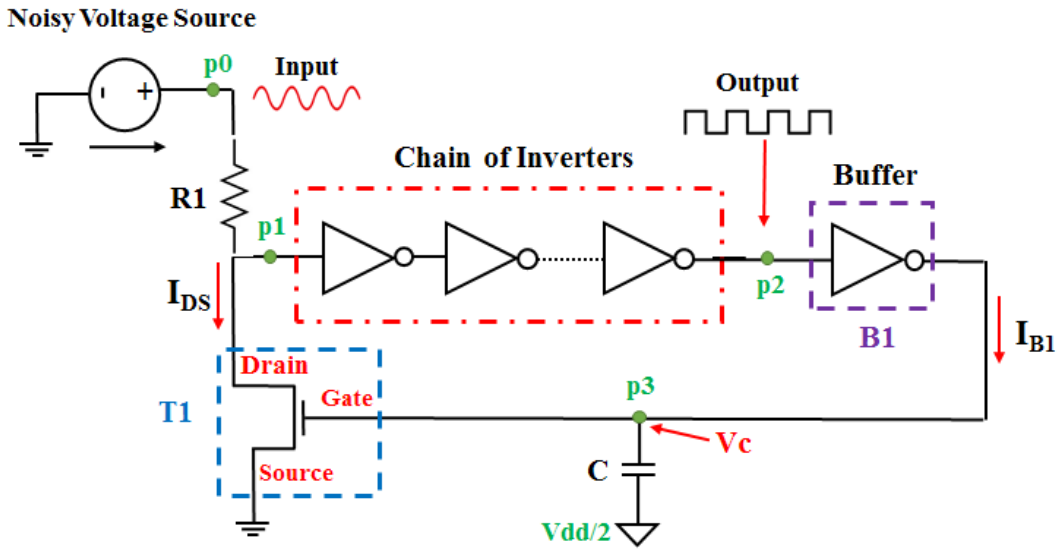


Fig. 4.7: DVFT circuit model for the proposed TRNG.

DVFT Circuit Operation

As shown in Figure 4.7, $p1$ is the point that the input voltage is applied to the chain of inverters. Depending on the variation of the input voltage at this point ($p1$) the TRNG circuit will no longer function correctly. If the voltage V_{p1} is higher or less than $\frac{V_{DD}}{2}$, in our case 2.5V, the feedback tuning system (i.e. DVFT) should be able to adjust for this change in voltage. If the V_{p1} is not around $\frac{V_{DD}}{2}$, the power supply input drifts, there are two scenarios that occur at point $p1$ during the operational lifetime of the circuit. We now examine the effects of these two different cases with respect to the operation of the TRNG system:

i) **Case1** ($V_{p1} > \frac{V_{DD}}{2}$): In this case, when the voltage at point $p1$ is higher

than 2.5V, more zeros will be generated than ones at $p2$. This will translate to more ones than zeros after the buffer at $p3$. Over time, more ones will gradually raise the mean voltage at $p3$ and charge the capacitor (C) to a value more than 2.5V. This charging of the capacitor increases V_{GS} (gate voltage of the transistor) as well as the value of V_{DS} (drain voltage of the transistor) which leads to an adjustment in the value of $p1$.

At equilibrium, we expect that $V(p1) = V_{DS} = V_{GS} = \frac{V_{dd}}{2}$. Thus, the transistor (T1) is working in *saturation mode*, because the drain-to-source voltage (V_{DS}) is greater than the gate-to-source voltage (V_{GS}) minus the threshold voltage (V_T) as follows:

$$V_{DS} > V_{GS} - V_T \tag{4.2}$$

Since T1 is in saturation mode then the value of the saturated drain current (I_{DS}) is given by the following equation:

$$I_{DS_{T1}} = \beta_{T1} \frac{(V_{GS_{T1}} - V_T)^2}{2} \tag{4.3}$$

where $\beta_{T1} = \mu C_{ox} \frac{W_{T1}}{L_{T1}}$.

If V_{GS} has been affected by the charging or discharging of the capacitor, then based on Equation 4.3, the value of I_{DS} is also affected. In this case (Case1), an increase in the value of V_{GS} causes the value of I_{DS} to also increase. As I_{DS} increases, the voltage drop across $R1$ increases, thus lowering the voltage at $p1$.

The net effect is that the original increase in the voltage at $p1$ has been counteracted, and the system returns to equilibrium with the voltage at $p1$ returning to $\frac{V_{DD}}{2}$.

ii) **Case2** ($V_{p1} < 2.5$V): In this case, when the voltage at point $p1$ is low, more ones will be generated than zeros at $p2$. This will translate to more zeros than ones after the buffer at $p3$. Over time, more ones will gradually lower the mean voltage at $p3$ and discharge the capacitor (C) to a value more than 2.5V. This discharging of the capacitor decreases V_{GS} (gate voltage of the transistor) as well as the value of V_{DS} which leads to an adjustment in the value of $p1$. Similar to the previous case, decreasing V_{GS} decreases I_{DS} which decreases the voltage drop across $R1$ thus raising the voltage at $p1$ and eventually returning the system to equilibrium.

DVFT Circuit - Capacitor Influence

In Figure 4.7, the capacitor (C) plays a key role in the overall operation of the feedback loop (DVFT) in the TRNG circuit. This capacitor is precharged to 2.5V, allowing it to maintain a fluctuation charge around the ideal voltage (2.5V) of the input ($p1$) to the inverter chain, as shown in Figure 4.8. The size of the capacitor determines how much time the feedback takes to react to changes in the voltage at $p1$. More specifically, it reacts to the average number of 1's and 0's at $p2$.

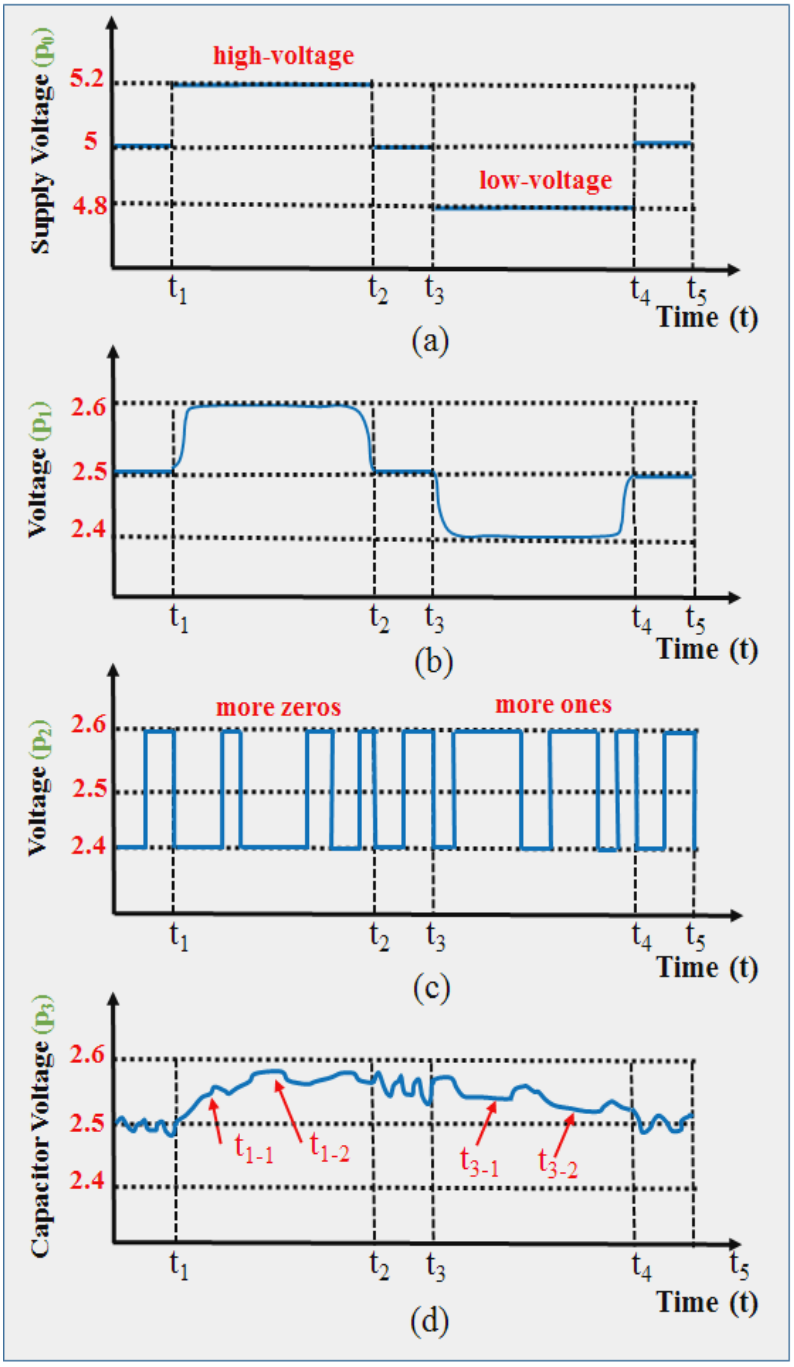


Fig. 4.8: Timing diagram of different points (p0, p1, p2, p3) of our model.

In order to determine an ideal size of the capacitor in our design, we look at the voltage and current levels of different parts of the DVFT design which is shown in Figure 4.8.

The capacitor is discharged and charged through the buffer $B1$, and the current, I_{B1} can be calculated as follows:

$$I_{B1} = C\frac{dV_c}{dt} \longrightarrow \frac{dV_c}{dt} = \frac{I_{B1}}{C} = \frac{dV_{GS_{T1}}}{dt} \tag{4.4}$$

where C is the capacitance of the capacitor and dV_c is the change in the capacitance voltage or also the the change in the gate-source voltage of the feedback transistor T1 ($dV_{GS_{T1}}$).

As mentioned earlier, T1 is in saturation mode and the derivative of the saturated drain current is calculated based on Equation 4.3. Thus:

$$\frac{dI_{DS_{T1}}}{dV_{GS_{T1}}} = k_{T1}V_{GS_{T1}} \tag{4.5}$$

Based on Equation 4.4 we know that $\frac{dV_{GS_{T1}}}{dt} = \frac{I_{B1}}{C}$. Therefore:

$$\frac{dI_{DS_{T1}}}{dV_{GS_{T1}}}\frac{dV_{GS_{T1}}}{dt} = \frac{dI_{DS_{T1}}}{dV_{GS_{T1}}}\frac{I_{B1}}{C} \tag{4.6}$$

$$\frac{dI_{DS_{T1}}}{dt} = \beta_{T1}V_{GS_{T1}}\frac{I_{B1}}{C} \tag{4.7}$$

We also know that the voltage level at point $p1$ is:

$$V_{p1} = V_{DD} - RI_{DS} \longrightarrow \frac{dV_{p1}}{dt} = -R\frac{dI_{DS_{T1}}}{dt} \tag{4.8}$$

We can then calculate $\frac{dp}{dt}$ which is the rate of change of the percentages of 1's over time, using both $\frac{dp}{dV_{p1}}$ and $\frac{dV_{p1}}{dt}$, because:

$$\frac{dp}{dt} = \frac{dp}{dV_{p1}} \frac{dV_{p1}}{dt} \tag{4.9}$$

where $\frac{dp}{dV_{p1}}$ will be calculated based on the slope of the graphs in Figure 4.6. Putting Equations 4.7, 4.8 and 4.9 together and substituting $V_{GS_{T1}} = \frac{V_{DD}}{2}$, we get:

$$\frac{dp}{dt} = -\frac{dp}{dV_{p1}} \frac{R}{C} \beta_{T1} I_{B1} \frac{V_{DD}}{2} \tag{4.10}$$

Using Eq. 4.10, we can derive the rate of change of the percentages of 1's in terms of the ratio, $\frac{R}{C}$,and the sensitivity of the the 1's percentage to changes in the supply voltage ($\frac{dp}{dV_{p1}}$). Thus, depending on how quickly we want the 1's percentage to react to changes in the supply voltage, we can select R and C appropriately. In Section "experimental", we show an example of how to select these values.

Figure 4.8 represents our capacitor waveform model when introduced to a discretized input voltage test pattern from a power supply. The purpose of this figure is to show the functional operation response of different labeled points (p0,p1,p2), ending with the capacitor(p3), as shown in Figure 4.7. The timing diagrams for each of these points (p0,p1,p2,p3) are represented by Figure 4.8 (a), (b), (c), and (d), respectively. We describe the timing and voltage behavior of

these points as follows:

- Figure 4.8 (a) This graph shows an example voltage test pattern for running this circuit, arriving in at point $p0$ from Figure 4.7. The important aspects of this figure are related to the voltage fluctuation, represented by a change in voltage from 5V to 5.2V (high voltage), or 4.8V (low voltage), and the time slots noted by t_1, t_2, t_3, t_4, and t_5.

- Figure 4.8 (b) This image represents the voltage present at point $p1$, from Figure 4.7, which is the input voltage to the inverter chain. Because of the voltage divider, one will notice that the voltage response range is from 2.4V to 2.6V. Along the time boundaries (i.e. t_1, t_2, t_3, t_4, t_5) the voltage response is no longer discretized since the input capacitance to the first inverter, of the inverter chain, causes a continuous behavior (charging/discharging) of the voltage level around this point ($p1$).

- Point $p2$ is the point where our TRNG output (generated bitstreams) are produced. As shown in the timing diagram Figure 4.8 (c), at the cases where the supply voltage has the ideal value (5V) one can see that the generated number of '1's (2.6V) and '0's (2.4V) are equal; these regions being: before t_1, t_2 to t_3, and t_4 to t_5. In the cases where the supply voltage fluctuates above or below the ideal case, there is an bias induced causing an increase

in the number of '0s' produced (t_1 to t_2) or the number of '1's generated (t_3 to t_4).

- Point $p3$ represents the capacitor's (C) voltage level; initially precharged to 2.5V. The job of this capacitance is to regulate the voltage of the circuit around 2.5V, as can be seen in time periods where there is equal generation of '0's and '1's (Figure 4.8 (c)). The areas of interest are when there is a bias in the generation of '0's (emphasized by t_{1-1} and t_{1-2}) or in the production of '1's (emphasized by t_{3-1} and t_{3-2}). In these regions one can clearly see the effect of the biases shown in Figure 4.8 (c) to the resulting waveform in Figure 4.8 (d). Since there is a buffer (B1) between points $p2$ and $p3$, the capacitor (C) charges in the presence of more '0's being generated and discharges when there is a bias towards '1's.

4.6.3 Data Analysis and Results

While we have presented what appears to be a solid and effective hardware RNG design, we can not stop our investigation of this schema purely on the theory behind its operation. We proposed a TRNG model that is simple, lightweight, and requires minimal additional area overhead. Now we need to examine the operation of this device under different conditions (e.g. voltage changes, changing current) to ensure that it remains robust and stable. One may be wondering: is

this circuit reliable and efficient despite being simplistic? Are the bits generated by this hardware RNG acceptable for use within critical designs? In this section we will examine our TRNG design to determine its robust behavior and function. We will begin by examining how the addition of the feedback loop aids self-adjusting nature of our proposed TRNG circuit. We will then examine the effects of using a programmed microcontroller (load) in order to see the effect of changing voltage (to the input of the inverter chain) on the operation of our entire TRNG circuit behavior. We will examine the generated output from the hardware RNG to ensure that it meets the NIST Test standard and, lastly, examine other possible avenues of producing bias in the generated output.

Feedback Loop Voltage Adjustment

As one can see from Figure 4.6, there is a clear advantage to including dynamic voltage feedback tuning (DVFT) to our hardware RNG design. Each of these graphs represent the use of different power supplies to introduce voltage into the same hardware RNG model, where the x-axis represents the input voltage level and the y-axis shows the percentage of '1's generated in the output based on the specific input voltage (V_{in}). With the blue dotted line (without implementation of a feedback loop), one can see that the desired voltage of 2.5V is a difficult area to maintain with minimal room for error during operation. However, the red dotted

line (including a feedback loop) shows a clear plateau around 2.5V allowing for optimal operation of our TRNG under a changing voltage. Figure 4.6 (b) shows that when using the Agilent E3630A (bench power supply), the percentage of '1's is approximately 20% when the input voltage is around 2.5V (in the case of no-loop). However, in the case of with-loop, that same power supply manages to maintain a percentage of '1's around 50% for an area of 0.5V around 2.5V. This greatly illustrates the advantageous behavior of including the DVFT to the TRNG model. Furthermore, in Figure 4.6 (e), one can see that when using the 06TM1C Dell (laptop adapter; functioning at 19.5V) that while the percentage of '1's produced at the output is near 50% at 2.5V (in the no-loop case), the margin of error is extremely small in comparison to the with-loop condition.

As shown in Figure 4.9, the percentage swing of the tunability range is approximately 20%. For sensitive circuitry this swing is appropriate as it would become damaged before the DVFT loop would become affected. With circuitry that can handle a larger swing (i.e. greater than 20%) then one solution would be to add addition inverters to the DVFT. By providing more inverters, a larger range is produced. Furthermore, one might notice that the percentages of responses are different among power supplies because the properties of power supplies are not identical that make them to react differently at various voltage levels. Note that the divergence in resistance, conductance, and capacitance of each power source

causes the difference in output.

Voltage Sweeping using Arduino Microcontroller

Here we examined the affect of replacing the input power supply voltage coming into the inverter chain with an Arduino Uno that is already being used to take readings from point $p2$ (shown in Figure 12). The purpose for implementing a programmed Arduino is that it can add a load to the input of the inverter chain while it is operating for reading the output of the TRNG circuit's bitstreams. To further cause fluctuation to the input voltage, a pull down resistor (R_{pd}) was added to the circuitry to provide more voltage change. In this scenario, we observed that the operation of our proposed feedback tuning circuit remains stable while applying a variable load (Arduino) to its input. This validates our model's use in hardware RNG applications under variable input voltages.

Randomness Tests Results

In the previous writings, we have shown that the hardware RNG is capable of adjusting to the voltage using our dynamic feedback tuning system. While this aspect of the TRNG model is functioning, one must be sure that the output generated by our model has enough unpredictability to allow its usage in critical applications such as cryptography, chaos theory, and key generation. We must apply tests to evaluate the randomness of the data to ensure that the gen-

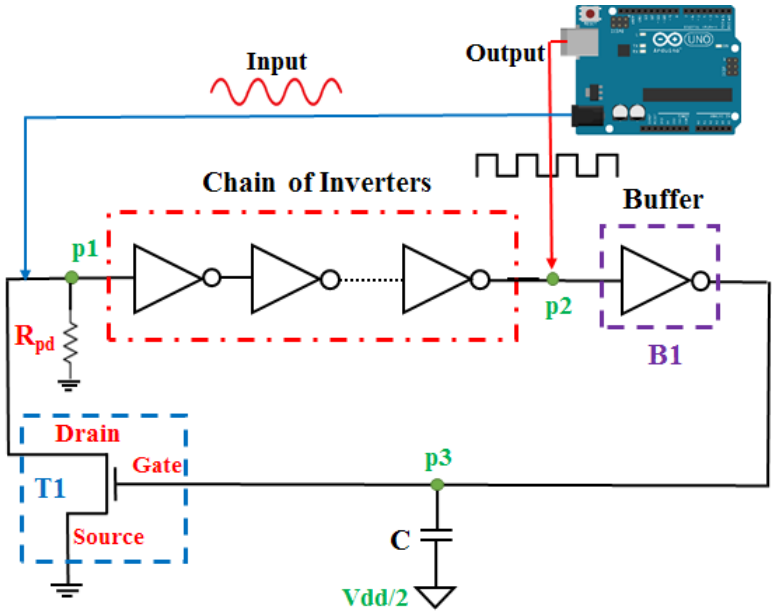


Fig. 4.9: Voltage sweeping using Arduino as a variable load to the input.

erated output (bit sequences) can be implemented and applied to areas where this random nature is a critical requirement. The NIST statistical test suite is a suitable metric that can be used to investigate the degree of randomness for generated binary sequences generated from a TRNG model. As mentioned in Section "namesection", the important aspect of the NIST statistical test results are the generated p-values. The results of NIST test for different power supplies are shown in Table 4.3; indicating that each power supply successfully passed the NIST tests.

Table 4.3: NIST Statistical Tests Suite average p-value results of our proposed TRNG model using feedback loop (DVFT) implemented with different power supplies.

NIST Tests	p-value **MASTECH HY3005F**	p-value **Agilent E3630A**	p-value **Agilent E3647A**	p-value **CX305N-00 Dell**	p-value **06TM1C Dell**	p-value **USB E238846**
Frequency	0.7419	0.8772	0.6403	0.7954	0.8802	0.7346
Block Frequency	0.7691	0.4720	0.7309	0.5932	0.8604	0.7759
Runs	0.7360	0.7155	0.6803	0.8522	0.5192	0.8061
Longest Run	0.4370	0.6182	0.7509	0.8023	0.5003	0.8207
Cumulative Sums	0.8920	0.7305	0.9137	0.9025	0.9234	0.8113
Rank	0.6954	0.7305	0.7844	0.4970	0.8364	0.5513
FFT	0.8567	0.7419	0.8320	0.8023	0.6591	0.7865
Linear Complexity	0.9075	0.8910	0.7745	0.9411	0.7904	0.8608
Overlapping Template	0.6433	0.6850	0.5374	0.5782	0.7129	0.5553
Non Overlapping Template	0.6631	0.7105	0.6925	0.4063	0.8653	0.6163
Approximate Entropy	0.8740	0.7801	0.9362	0.9120	0.9544	0.8561
Universal Statistical Test	0.4173	0.6482	0.7305	0.6682	0.5492	0.4318
Random Excursions Variant	0.6281	0.5836	0.7015	0.6637	0.8630	0.9043
Random Excursions	0.8036	0.8137	0.7382	0.8390	0.7734	0.6204
Serial	0.9213	0.9473	0.8129	0.9552	0.9352	0.8941

Bias Influences

Bias test is a check for potential operation influencing variables that can cause the generated output of our design to become biased toward producing a greater percentage of '0's or '1's. These influencing variables can come in a larger number of forms, ranging from physical properties of the circuit elements, environmental changes during normal operation, or even the affect of prolonged operation over a large amount of time. When attempting to force bias into the system, we found that our proposed model (DVFT) ensured the desired bitstream output (e.g. equal number of '0's and '1's) even when being introduced to a variable load into the inverter chain. Our belief is that the most influential effect on our TRNG model is the introduction of a sweeping input voltage, which we were able to show could be overcome by our DVFT loop.

4.6.4 Attacks and Defenses

Since TRNGs are widely spread within critical applications, their security and trustworthiness must be validated to ensure their functionality has not been tampered by adversaries. These components are of great interest to malicious parties because if these single elements can be compromised with pseudo random behavior then their output can be predicted in a deterministic manner. When dealing with the attacking of our TRNG model, depending on how the circuit is implemented

changes the potential attack surface. The two implementations are: on-chip and off-chip design. Also, depending on how one implements these two solutions, the attacks [67] performed could be categorized as invasive [112] or non-invasive [66]. Invasive attacks change the functionality of the TRNG by modifying the actual circuit or design of the hardware RNG which leads to a permanent effect to the entire TRNG system [65]. Non-invasive attacks are those that do not permanently alter or deform the function of a TRNG. These types of attacks focused on influencing the input or environment in which a TRNG operates. We tackle theses concerns in detail in the following subsections.

On-Chip Design

An on-chip design is where the entire TRNG circuit and power supply are part of a single integrated circuit. This method has its own properties that can cause an IC design to be more favorable while also introducing additional design complexities. When creating an on-chip solution there are various aspects that one must consider. Complex impedance profiles from 14 metal layers, complex workloads due to other circuitry or applications running on the device, multi-frequency droops due to IR, inductive, and capacitive properties. Furthermore there is the concern that depending on the workload of other circuits, this could cause changes to the power supply signal. One can overcome these aspects by adding overhead like

guard-banding for a trade-off of additional overhead cost. This overhead causes the the solution to no longer be lightweight. One might wonder what the potential for a malicious individual to introduce power supply fluctuations into the TRNG model. One method of addressing this problem would be to have another circuit element to detect the presence of biasing in the output of our TRNG. It is worth noting that at this point, since the power supply would be part of the IC, performing a frequency injection attack [29] would classify as an invasive attack. Another method involves moving the power supply off chip, however this introduces easier access to tap the power line. One could have an isolated IC for the TRNG circuit, which introduces the need for a dedicated power pin. The use of a dedicated power pin makes the attack vector of signal injection easier to perform. The use of this power pin does change the nature of a frequency injection attack from being invasive to a non-invasive one. To overcome the power pin limitation, one could use the output of the TRNG into a Line Feed Shift Register (LFSR) as a seed for creating required random numbers. This would add another layer of randomness because although one may modify the seed using the outside pin, the physical variations of the chip itself would not allow one to be able to predict the pattern.

Off-Chip Design

An off-chip design is where all the components in our circuit, plus the power supplies, are implemented on a PCB rather than in IC. Integrating the components on PCB has different advantages and disadvantages relating to the implementation and use of our TRNG model. This can allow for the lightweight nature of the circuit; under the assumption that the PCB is secure and can not be tampered with. If this assumption can not be made then the lightweight nature of the off-chip solution raises new attack concerns. When creating an off-chip solution the main concern is that it causes most attacks to go from being invasive (in the on-chip model) to non-invasive and therefore easier to implement. One might be concerned about the ability to perform a body effect or back gate attack that could cause biasing of the TRNG model. Since the circuit is not as well protected, then it would be easier to manipulate the regions of the transistor's operation. The purpose of the DVFT element is to remove this bias and allow for random output from the circuit, even if it requires a single cycle. The greatest concern in creating an off-chip solution is the increased access to the external power supply. This attack vector is a prime candidate for performing frequency injection and altering the output of the TRNG. As with the on-chip solution, one could integrate a detection element to notice the presence of biasing in the output of the TRNG. However this solution does generate greater overhead effecting the lightweight

nature of our design.

In general, for both the on-chip and off-chip solutions temperature can also effect operation of the DVFT. So long as the voltage stays within the specific range of operation then temperature fluctuations are acceptable. Should the voltage move outside of this range, then the DVFT will no longer work.

4.7 Implementation of Power Supply-noise based TRNG in Linux Operating System

We also considered the construction of hardware and software implementation of the True Random Number Generator (TRNG) that inputs data into the Linux operating system. The hardware interface is to convert the input noise from power supply into sequence of bits, which is then read by software. The software interface connects with TRNG driver, Linux kernel and framework services to pass data from TRNG driver to Linux kernel. The final testing consisted of NIST STS of our implemented system. The testing consisted of measuring true randomness of bits, before they are provided to the Linux OS. Based on our NIST test, if our P-value is greater than 0.01 (minimum value), we can conclude that data is 99 percent random.

4.7.1 Functionality of the Proposed TRNG Model in Linux

The feasibility of using power supply noise as a source of entropy is tested by implementing the TRNG on an embedded Linux system and auditing the random numbers that the TRNG driver supplies to the system PRNG. Power supply noise is processed by attenuating it to produce a mean voltage equal to the transition voltage of a logic inverter. This attenuated signal is supplied as an input to a chain of inverters. Each inverter amplifies the power supply noise, eventually producing a sequence of logic low and logic high signals. This sequence is sampled by a buffer and this buffer is read by the TRNG driver. The random numbers produced by the TRNG driver are audited using the NIST Statistical Test Suite. The results of the NIST Statistical Test Suite show that the noise produced by a power supply can be used as an entropy source for a random number generator implemented on an embedded Linux system.

4.7.2 Experimental Setup

In order to fully optimize our system, selecting the best hardware and software options by balancing performance and user friendly was absolutely necessary. The first step of implementing the TRNG was to determine appropriate hardware to run Linux on. Initial research suggested that a BeagleBone, a Xilinx Nexys 4 DRR FPGA, and a Raspberry Pi would be suitable to implement the TRNG on.

Hardware choices were compared by the maximum bit-rate an input pin could be read by software, the amount of time required to get a Linux based operating system running on the hardware, the cost to acquire the hardware, and the time required to obtain the hardware.

4.7.3 Randomness Test

The NIST Test Suite is a statistical package consisting of 15 tests that were developed to test the randomness of (arbitrarily long) binary sequences produced by either hardware or software based cryptographic random or pseudorandom number generators. These tests focus on a variety of different types of non-randomness that could exist in a sequence. Note that the 15 tests are as follows: Frequency Test, Block Frequency, Runs, Longest Run, Cumulative Sums, Rank, FFT, Linear Complexity, Overlapping Template, Non-overlapping Template, Approximate Entropy, Universal Statistical Test, Random Excursions Variant, Random Excursions, and Serial. The results of NIST Statistical Test suite is presented in Table 4.4.

4.8 Conclusion

This chapter proposed a novel hardware RNG model that makes use of power supply noise to generate random bit streams that can be applied in critical ap-

Table 4.4: NIST Statistical Tests Suite Results

NIST Tests	P-value	Result
Frequency	0.213309	passed
Block Frequency	0.122325	passed
Runs	0.534146	passed
Longest Run	0.739918	passed
Cumulative Sums	0.376154	passed
Rank	0.534146	passed
FFT	0.739918	passed
Linear Complexity	0.122325	passed
Overlapping Template	0.534146	passed
Non Overlapping Template	0.502086	passed
Approximate Entropy	0.534146	passed
Universal Statistical Test	0.350485	passed
Random Excursions Variant	0.140647	passed
Random Excursions	0.159354	passed
Serial	0.122325	passed

plications. This design is a lightweight solution using few electronic elements in order to construct our TRNG circuit model, which adds a minimal amount of area overhead to the entire system allowing it to be very simple in comparison to other existing TRNGs. The proposed system also implements our new dynamic voltage feedback tuning (DVFT) circuitry to allow for robust and effective behavior in the face of voltage and current fluctuations; creating self-adjusting operation over time. Our results, based on experimenting with the use of six different power supplies (three bench power supply, one computer power supply, one laptop adapter, and one USB), show that our hardware RNG is capable of regulating its operation under various conditions (e.g. sweeping voltage, current changes). Our NIST results show that in each of these scenarios (e.g. different power supplies) the TRNG model passed all 15 tests, which indicates that the generated bit sequences are random. In future work, we intend to examine the operation of this model under various additional environmental and operational conditions, such as temperature variation and aging, to determine the ability and robustness of our work in different circumstances.

Chapter 5

Hardware Security Architecture for the Internet of Things (IoT)

5.1 Introduction

The Internet has drastically changed the way we live, moving interactions between people at a virtual level in several contexts spanning from professional life to social relationships. The origin of the Internet of Things (IoT) can be traced back to the development of the Internet (interconnected network of computer networks). IoT is a novel paradigm that is rapidly gaining ground in the scenario of modern wireless telecommunications [7]. Another basic property of these things is push button connectivity to the Internet or peer devices. IoT is comprised of a number of technological protocols that aim to make up connections from one object to other things and databases. The reason that one would to connect IoT devices is to exchange information and monitor data which needs to be secured. The first place where trust needs to be established in at the hardware level which is the

platform that any software runs on top of. IoT still has many challenging issues that need to be addressed and both technological as well as social knots that have to be untied before the IoT idea becomes more widely accepted.

In this chapter, we examine the applications of hardware-based random functions security primitives for the purpose of establishing trustworthy hardware within any authentication framework. In order to do this, we evaluated two case studies (Case Study 1 and Case Study 2) to determine the feasibility of our solutions. Case Study 1 looks to provide authentication to an unknown number of hardware platforms over an untrusted network in IoT healthcare domain. Case Study 2 analyzes the possibility and cost of implementing our hardware security primitives into an existing smart IoT device.

5.2 IoT Adoption and Expansion

The 'Internet of Things' and IoT devices have been on the rise as of late due to a shift towards more 'cloud-centric' models of interconnected devices performing actions or integrating with the everyday life of users. Over time the hype and expectations of IoT technology have changed, but the overall desire to have a more vibrant and interconnected 'smart' Internet still runs deep in the human consciousness [32]. As one can see in the Figure 5.1 (Source: Gartner Inc. [58]), the Internet of Things will continue to grow and develop, all while influencing

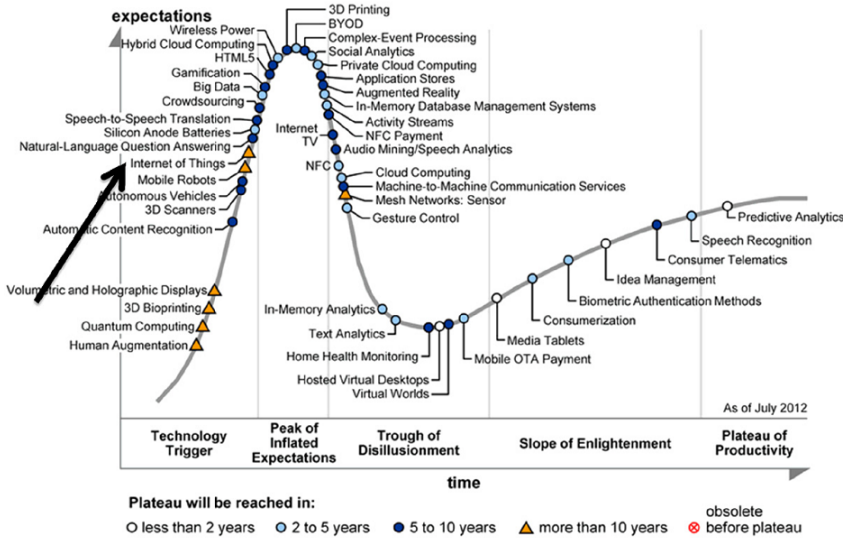


Fig. 5.1: Gartner 2012 Hype Cycle of emerging technologies.

IoTs domain by providing new evolving data and the required computational resources for allowing users and developers to create revolutionary applications [32]. With this expansion of use cases and implementations comes a requirement to examine the security needs and specifications that can secure the new flux of information and data from prying eyes or malicious action. In order to best protect this expanded adoption of IoT, our solution is to secure the IoT hardware using hardware-based random functions.

5.3 Security and Privacy Challenges in IoT Systems

IoT is generally a large number of wireless devices that form a network. The resulting 'Internet of Things' is as powerful as it is susceptible to the same vul-

nerabilities and security flaws as any computer system or distributed system of computers. Securing any stored data, ensuring access control to sensitive or critical areas of function, encrypting communications channels and authenticating new/connected devices are all aspects of IoT security that must be taken into consideration when designing everything from a single IoT device to a distributed network of IoT devices. Security considerations for IoT devices are the same as those required for distributed systems or embedded devices. One must secure not only the information being interacted with on an internal level, but also ensure that any data/information exported by the device must maintain a level of security assurance desired by the developer and user. We have focused on improving hardware level security using hardware-based random functions primitives to establish trust for users and devices. Work by West et al. [103] has taken an in-depth analysis of the complications and errors that occur when security is implemented in fitness tracking IoT devices, along with a detailed postulation on how to suitably implement security policies and principles in an all-encompassing method. The similarity between our work and others in order to provide security and assurance for hardware is that there is cost and overhead for implementing these solutions and frameworks.

Common erroneous implementations of security lead to issues ranging from denial of service vulnerabilities, falsification of data (both local and remote users),

stealing or abuse of sensitive information, compromise of device integrity, or as simple as incorrect handling of shared data leading to sensitive information being leaked. A developer's focus can be placed at a variety of abstractions, all with the intention of making a device, or larger system, more secure and trustworthy. Beyond the security concerns, embedded systems have greater constraints in system design then other computer systems. Being an embedded system, the nonsecurity considerations boil down to power consumption, total PCB space, heat distribution, production costs, and component operation conditions. Depending on the degree of security desired and the budget allotted for development of a device, the overhead of incorporating security can become costly, thus making developers to ignore improve security in their hardware platforms. All of these different aspects play into the constraints and optimization of designing any secure embedded/IoT device [5]. Our methods for securing IoT devices are focused on implementing low-cost techniques in such a resource constrained devices (IoT devices).

5.4 From Hardware Security Primitives to the Internet of Things

Hardware security primitives are basic components or elements that can be used to construct a more secure and sophisticated platform. They are used to provide security for a device using the physical properties (process variations) during manufacturing and fabrication processes of the device. The advantage of using

these security primitives is that they are very low-cost solutions in terms of cost and overhead since they require minimal resources to function. As mentioned in the previous chapters, Physical Unclonable Functions (PUFs) and True Number Generators (RNGs) are two important hardware security primitives that could provide a solution to securing hardware platforms that are used in IoT networks. The reason that developers should protect the hardware is that physical attacks are common since hardware attack is the first and easiest step. Furthermore, if an adversary gains control of the hardware platform, then they will be able to access any software running on the device. One way that this could manifest is if an IoT device is using human characteristics (Biometrics) to authenticate users, then a malicious actor that has control over hardware could see the sensitive data passing through to the software. Our solution to this issue is to verify that the hardware is not tampered using hardware security primitive methods, thus ensuring that the human characteristics authentication is a trustable. Note that human characteristics based identification and authentication has been proved to be unique for each individual and can be used to authenticate users with a high degree of accuracy. Since human characteristics based solutions are favorable for people to uniquely identify themselves, this form of authentication is highly desirable for IoT devices. Because people want to use their biometrics for authentication, one can expect a large amount of adoption for this authentication method as the num-

ber of IoT devices continues to grow. The reason one needs to protect hardware is that any authentication solution, at some point, will be executed on top of a hardware platform.

Therefore, two case studies that have been evaluated in this chapter examine the high level use of hardware security primitives to securely authenticate IoT systems using PUFs and biological signals.

5.5 Case Study 1: Our Proposed Architecture for Authentication in IoT Healthcare Applications

One important application of the IoT is in the healthcare domain. There are benefits provided by IoT technologies to the healthcare and the resulting applications can be grouped mostly into: tracking of objects and people (staff and patients), identification and authentication of people, automatic data collection and sensing [99]. The importance in establishing secure hardware platforms for each IoT device in a monitoring network is that devices (e.g. Wearable IoT) need to be verified because they might act maliciously based on counterfeiting hardware. Furthermore, the user (patient) should be trusted within a network where data is being exchanged.

5.5.1 Infrastructure Security In Healthcare

In the Internet of Things in the health care domain, a patient will not necessarily have to travel to see a doctor anymore. Since he or she can upload the collected vital signals from sensors to the cloud from smartphones, tablet computers, or home remote controllers for a doctor or trained specialist, this data can be processed by signal processing analysis, through the cloud systems in real time across large sets of any patient supplied data. Moreover, by analyzing the patient's vital signals in real time, one can use this analysis to help identify and predict unhealthy biological signals like heart attacks for cardiovascular patients, etc.

Wearable devices are now at the heart of just about every discussion related to the IoT; a hallmark of the Internet of Things and the most ubiquitous of its implementations to date. Health and fitness oriented wearable devices that offer biometric measurements such as heart rate, perspiration levels, muscle activity, and even complex measurements like oxygen levels in the bloodstream are also becoming available. As wearable IoT use grows and the number of biological signals being recorded and processed increase, one needs to account for the security and privacy concerns of the individuals using the wearable technology. Risks include possible harm to the patient's safety and health, loss of protected health information and unauthorized access to devices and exploits of vulnerabilities. Therefore, an authentication framework for IoT is a necessary requirement in any

information system to ensure the availability of information to authorized users only. The best way to ensure patient are correctly authenticated is through the use of hardware-based random functions that provide unique identification and verify trust throughout the network of wearable monitoring IoT devices.

5.5.2 Motivation and Challenges

IoT in healthcare can be used to create a home monitoring system for elderly care, which allows doctors to monitor patients and the elderly in their homes, thereby reducing hospitalization costs through early intervention and treatment. With a system of this connectivity and size, there is a requirement of integrating security and trust into the exchange of currently in-alterable information. Unfortunately, IoT is extremely vulnerable to attacks for several reasons. First, physical attacks to the unattended components are easy. Second, since most of the communications are wireless, eavesdropping is extremely simple. Third, IoT components can not implement complex schemes to support security since they are characterized by limited resources and capabilities. This chapter presents a investigation of the infrastructure of IoT devices in their application to authentication within the healthcare domain. We investigate the issues of trust and trustworthiness within an environment of unknown devices used by unknown individuals over potentially unsecured networks in order to prevent the leaking of sensitive information of the

patients.

5.5.3 Proposed Security Approach for Authentication in IoT Healthcare

Figure 5.2 is a visual representation of the potential connectivity of IoT-based ubiquitous healthcare devices. One will notice that at the center of this mapping is a common cloud where these devices exchange information for the purpose of storage, authentication, or for medical observation. As one can see, there are varying solutions ranging from the type of protocols used to hardware solutions that provide effective and efficient security and privacy. The authentication mechanisms typically are using passwords, smart cards, tokens, and biological signals that are applied to reduce incidents harmful to patients including wrong drug, dose, time and procedure. The biometric features of every human are discriminatively identifiable, non-transferable, and non-reproducible, so why wouldn't a medical facility use these signals for identification? Especially since they are already collecting this same information from patients. Using wearable sensor nodes for human monitoring has its own vulnerabilities since the devices may leak personal information [4]. One must address the limitations and security concerns of the underlying hardware to be sure that both patients and staff are equally protected from malicious, or erroneous, behavior.

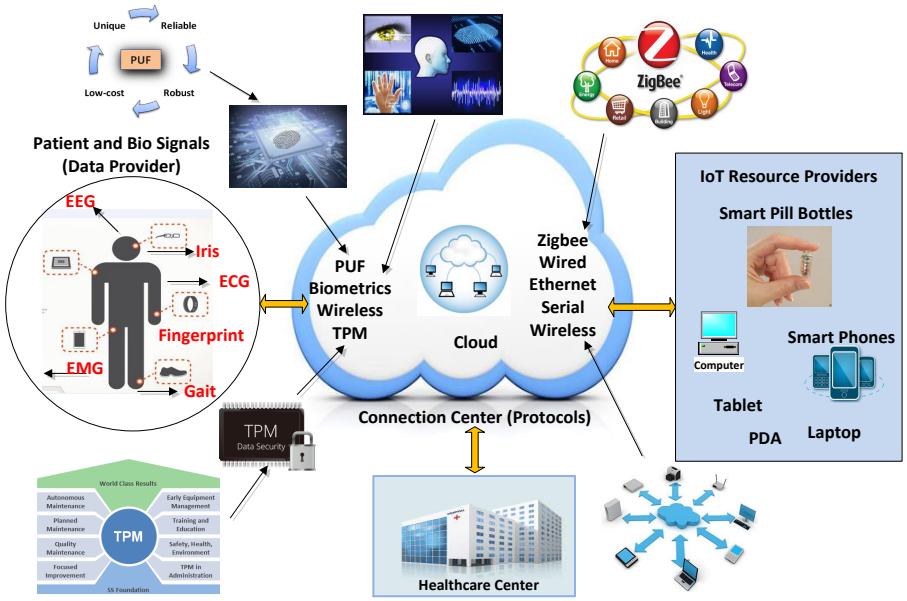


Fig. 5.2: A platform of IoT-based ubiquitous healthcare solutions using secure hardware elements for Authentication.

Device Verification

The first step that a developer needs to take when designing a monitoring network is to verify the hardware of devices in order to detect the presence of counterfeit and malicious functions. Here, we examine the use of PUF for establishing device verification through the use of hardware-based security primitives.

(1) PUF for IoT Applications: Counterfeit hardware is a problem that plagues not only larger data mainframes, but also is equally dangerous in the embedded systems market [14]. Generally the concerns center around security and how to detect hardware Trojans or other malicious 'motherboard' behavior. Other

concerns include the use of non-standard materials that can cause the fraudulent devices to produce incorrect signals or readings during its operation. One method of working around this would be to use already obtained 'trusted' devices already obtained and alter their function to include security. We can use some of the existing components in the wearable devices to make a security layer on top of the system gathering signals from a patient. But as any security professional will tell you, one can not plan for security as an afterthought. Therefore, there is a need to authenticate that a given piece of hardware is 'trustworthy'.

(1) PUF-based Authentication: Physical Unclonable Functions (PUFs) are one of the hardware solutions commonly used to authenticate with any given device. Although there are various types of PUFs, we focus more on memory based PUFs due to the availability of the memories in almost every embedded device [40]. Memories inside the wearable devices can be used for authentication purposes. The cost of including PUFs into existing hardware is relatively small. This technology allows for low-cost authentication of embedded system elements while generating volatile keys for cryptographic applications.

In our infrastructure any medical IoT devices used must first be registered with the hospital (acting as the trusted party mentioned earlier). When a patient comes to purchase a medical IoT device from the hospital's stores, they will perform a device enrollment that will allow the hospital to associate the given

device with the specific patient. Once finished with enrollment, the patient will go home where they will activate their medical IoT system. This system will send a PUF response to the hospital database that the hospital will verify to their enrollment database. Once validated, the hospital will send an enabling signal back to the patient's medical IoT device allowing it to read their biological signal, thus authenticating the device within the medical IoT network. The patient authentication aspect will be described in the next section. With an expected use of millions of healthcare IoT devices there is a clear need for being able to authenticate these devices as trustworthy for healthcare related applications. Since these devices will already have some memory along with individualistic deep-submicron process variations, we propose that PUFs present a simple, low-cost solution for providing a base trust for staff and patients using these IoT devices. Once one trusts the operation of a given IoT device, it can be used for accurate reading of biomedical modalities which can be used to authenticate individuals from a pool of unknown users.

Patient Authentication

Healthcare applications are a very necessary part of our lives, providing the best medicals facilities to people suffering from various diseases. Hence, it is necessary for the hospitals to keep track of its day to day activities and the records of the

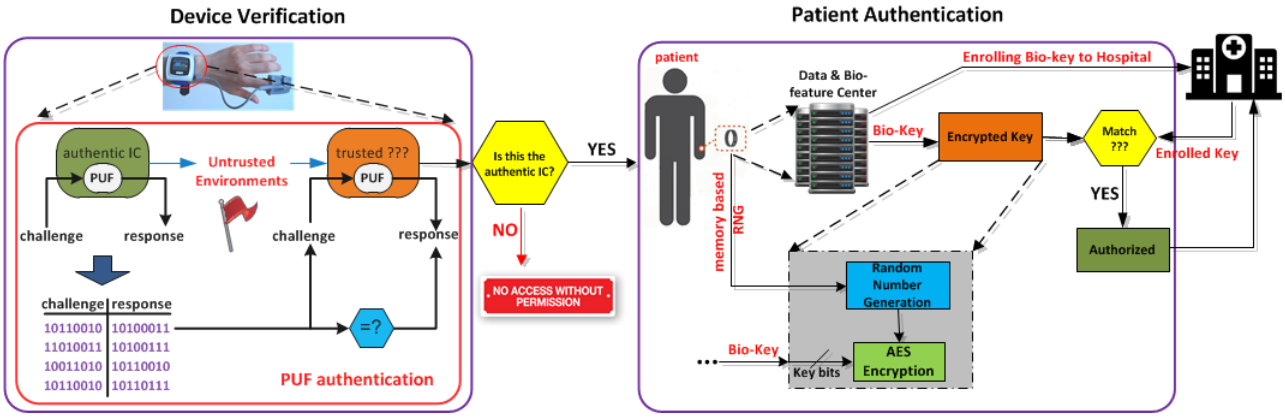


Fig. 5.3: Security Approach for Low-Cost Verification of Devices and Patients in the healthcare domain.

incoming patients. Monitoring patients remotely, collecting data and protecting them from tampering is very crucial to address security and privacy concerns. Although there are many benefits of healthcare application, there has always been a threat of hackers getting access to the sensitive information of the patients that can make misuse of that information leading to problems. To reduce preventative caution and impact level of the risk factor, we have investigated human characteristics-based authentication. human characteristics-based authentication/identification typically refers to utilizing interior and exterior physiological characteristics of a person (e.g. EEG, ECG, PPG, PCG, iris, fingerprint). The motivation for the extension of biosignal is in their potential uniqueness, universality, and their resistance to spoofing.

Electrocardiogram (ECG) signal not only is used for the purpose of diag-

nosis and treatment, but also can be used for biometric authentication and key generation [47]. Heart signals are one of the popular methods for use in authentication systems. This solution has its own advantages in terms of cost, ease of measurement and power. As shown in previous work [51], it is possible to use heart signals to generate a key that can be used for authentication purposes. The key generated from the ECG features was 727 bits in length, has an average entropy of 0.9810, and passed all the 15 NIST statistical tests. Note that this algorithm requires filtering, processing, and quantization of features as overhead for generating an authentication key. Further details of the process can be found in the previously cited paper. The authentication process requires an enrollment of an individual's ECG features into a database that can later be accessed to positively identify a user. Due to the use of already collected heart signals, this is a low-cost solution for dealing with authentication of unknown users within a larger healthcare network.

(1) Human characteristics for IoT Applications: Biometric authentication, identification and key generation systems have assumed increasing importance in recent years. There are two types of human characteristics methods that can be categorized into internal and external physiological traits of humans. Each of the biometric modalities, including fingerprint based and iris based approaches, exhibits particular strengths and weaknesses. Fingerprints are very popular due to

their low-cost implementation and well-developed feature extraction approaches. Iris identification is acclaimed for its high-level security, providing unique features even for identical twins. Even though these kind of biosignals are common, they are easy for attackers to access and are not robust against cloning. For instance, our fingers are involved in many daily tasks such as touching keyboards and door-knobs and can be easily replicated to bypass biometric systems. Iris systems are susceptible to being spoofed by printed photos. They are also expensive to implement.

In this chapter, we introduce the use of a new biometric modality known as the electrocardiogram (ECG) signal. The ECG modality is permissible given that it addresses the problems encountered by other biometric authentication research. These heart signals can be found in virtually all living humans (Universality). The authentication capabilities of ECG signals for circumscribed groups of individuals has been shown (Uniqueness). ECG signals can be easily acquired using suitable devices (Measureability). Electrocardiogram modality has been shown to perform accurately for subsets of the population (Performance) [71], [72]. The "off the-person" approach has made use of ECG signals acceptable (Acceptability), and they are not easily spoofed as it depends on an internal body organ, the heart (Circumvention). It also has inherent real-time signs of liveliness, making it extremely difficult to steal and emulate a persons ECG signal.

Electrocardiogram (ECG) [47], Phonocardiogram (PCG) [8], and Photoplethysmogram (PPG) [48] [50] are cardiovascular biometrics that are emerging as interesting choices for biometric systems that are internal physiological signals. Based on [47], bioelectrical signals recorded from the heart (electrocardiography, ECG) are distinctive enough for each individual person to be used for biometric applications, with the additional bonus of being inherently difficult, though not impossible, to forge. Also, they can be measured using low cost devices. Unlike other biometric systems, ECG signals can be monitored for prolonged periods of time: for example to continuously authenticate the user of a protected device after initial authentication. The Apple Watch applies the same principle of continuous monitoring, requiring the user to authenticate their identity with a password when the watch is strapped to their wrist, but then monitoring for constant heartbeat to avoid the need for further authentication. As additional security, once there is an interruption in the heartbeat detected by the watch, the watch locks itself down.

(1) Biometric-based Authentication: Our solution for IoT is a system that uses individual's biosignals for authentication. Human characteristics based method is a simple access control for an IoT healthcare device (wearable) that a person can use any fitness tracking IoT device, such as 'nymi', to access. To do so, first, all the subjects need to be enrolled. The enrollment phase contains biometric fea-

ture extraction, feature selection and finally the information of the helper data; as shown in Figure 5.4. The helper data contains the number of bits for each feature that can be quantized, the parameters of the boundaries, margin and the index of the features used for a given user. During authentication, an enrolled user supplies an ECG signal to the biometric system. The signal is preprocessed and features are extracted. The helper data is used to select the reliable features and quantize them to form the key. The key can be used to authenticate the individual by comparing it to a template. In a general case, the hardware requirements for a biometric authentication device would be the same, if not similar, to that of any standard 'smart' authentication device. The generated key could be stored in some temporary memory that could then be used for encryption or for the creation of a generated user identification certificate. This information would then be generated using the ECG data sent to the authenticating IoT device, which in turn would use this information to perform access control. Our security approach for authenticating only allows a system to operate when the correct biometric is presented, thereby protecting it against unauthorized access, as shown in Figure 5.4.

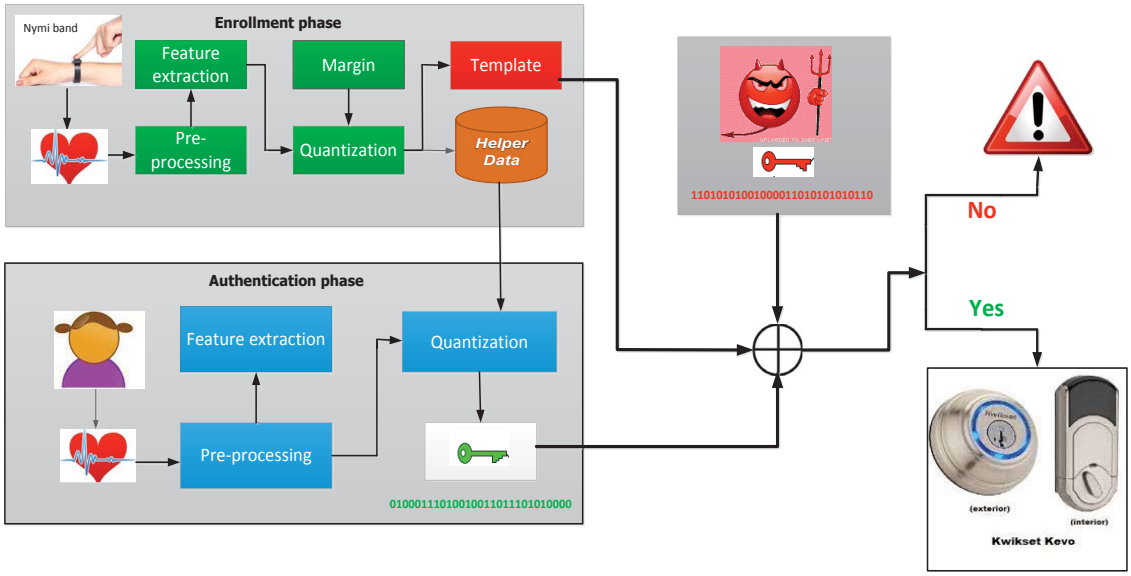


Fig. 5.4: Schematic for key generation procedure.

5.5.4 Examining Security of our Architecture Model

An overview of our security approach is illustrated by Figure 5.3. This approach is split up between a device verification step and a patient authentication step. We show that initially there is a device verification step where the healthcare IoT devices trust is established. Making use of existing memory on the IoT device, a DRAM-based PUF challenge-response pair can be used to constitute the trustworthiness of IoT operation. Should this verification fail then the device can not be trusted and thus will not be used for reading biological signals. This eliminates the concern that this IoT device would not operate as required in healthcare service which is responsible to verify the IC. The authenticated IC can

then be used to read the biological signals and perform the necessary processing to produce a bio-key. This bio-key is then enrolled to a local hospital allowing for later authentication of a given patient to the medical staff. This step, of course, requires encryption techniques. Note that the ECG could be used to generate a session key for encryption of the data in transit. However this aspect of the security approach is considered too broad for the scope of this chapter.

5.5.5 Case Study 2: Examination of using Human Characteristics (Biometrics) for Authentication in a Smart DoorLock IoT Device

In this case study, we move from examining the larger security framework presented in Case Study 1 to evaluating the cost and feasibility of implementing our PUF and human characteristics based authentication scheme on top of existing smart IoT device (Smart DoorLock). We have found that minimal additional overhead and cost is required to implement our design.

Motivation and Challenges

The exponential growth in the number of connected smart devices, and the resulting volumes of data, pose significant challenges for information security. Most cryptographic primitives rely on the ability to generate, store and retrieve unique keys. These cryptographic keys (unencrypted) are used as an input to the known

encryption engine to generate encrypted output that is used to authenticate the device or information. A shared cryptographic key enables strong authentication. Candidate sources for creating such a shared key include biometrics and physically unclonable function (PUF). However, maintaining large databases of PUF challenge response pairs and dealing with PUF errors makes it difficult to use PUFs reliably [109]. The use of biometrics has been widely spread over the problems of identification, authentication, and key generation [23], [93], [71], [10]. Most of the biometric authentication research is suffering from issues of universality, uniqueness, measurability, acceptability, and circumvention characteristics.

Architecture Hypothesis: IoT Device

As mentioned before, the concepts of an Internet of Biometric Things (IoBT) can be used in a wide variety of embedded applications that are connected to the Internet. Here, we specifically describe the architecture of an IoT device (Kwikset Kevo Smart Door Lock) in detail to show that it is a very suitable device (embedded system) to handle biometric authentication to become a more secure and reliable system. As shown in Figure 5.5, a generalized architecture of a 'smart' door lock consists of a microcontroller that contains a core processor, memory as embedded storage and input/output peripherals. The user input comes either from biometric readings (fingerprint, iris, ECG signals, etc) that are shown

in Figure 5.5, digit codes for use with the keypad (passwords), or some other form of wireless communication of data (e.g. Bluetooth, Zigbee, WiMax).

The lock mechanism is the common element of any door lock. The wireless communication of the Kwikset door lock can vary depending on user needs. In one variation of Kwikset's smart door locks, it was found that the wireless communication module, as seen in the right half of Figure 5.5, used the Zigbee protocol to interact with users. Another door lock variation replaced the Zigbee communication sub-module with a Bluetooth variation. The advantage of this modular behavior is that regardless of the communication protocol used, a developer only has to create a new sub-module rather than an entire PCB. This submodule connects to the larger door lock PCB through a communication channel (serial, I2C, etc.) that allows for exchange of data between the wireless antenna and a mixed signal microprocessor. The processor itself contains embedded flash memory. This separation of functional components and those responsible for 'external communication' shows the inherent design for resilience and adaptability as the needs and requirements of the smart door lock change over time. The left half of Figure 5.5 shows a simplified diagram example of Kwikset's door lock architecture and is meant as an easy to follow guide for continuing our discussion on the advantages of IoBT implementation [1].

Further investigation of the device's hardware revealed that the Zigbee pro-

tocol was used for communicating with the smart door lock device for an exchange of a traditional password/PIN. Surprisingly, there was no acknowledgement message sent back to the user interaction device confirming reception of the correct password/PIN. Upon realizing that no confirmation response was being transmitted by the smart door lock, we were able to determine that the authorized password/PIN combinations were stored locally on the door lock using the embedded storage. By no means is this the only implementation that can function for a door lock device; it is equally possible to have the door lock communicate with a remote database server for authentication or authorization needs. Through our examination of a smart door lock's architecture we have proposed that embedded devices have the opportunity of being more secure even in applications that are very crucial; home security devices, military applications, healthcare, etc. This does not mean that the other embedded devices are not eligible being used in IoBT concepts. As illustrated, although a subset of IoT devices may have been more advantageously designed for alteration, the design space for embedded real-time devices is vast and can easily be explored to produce a greater swath of modular IoT peripherals. We foresee a larger user adaptation of biometric authentication across a plethora of private and public sector implementations of IoT methodology.

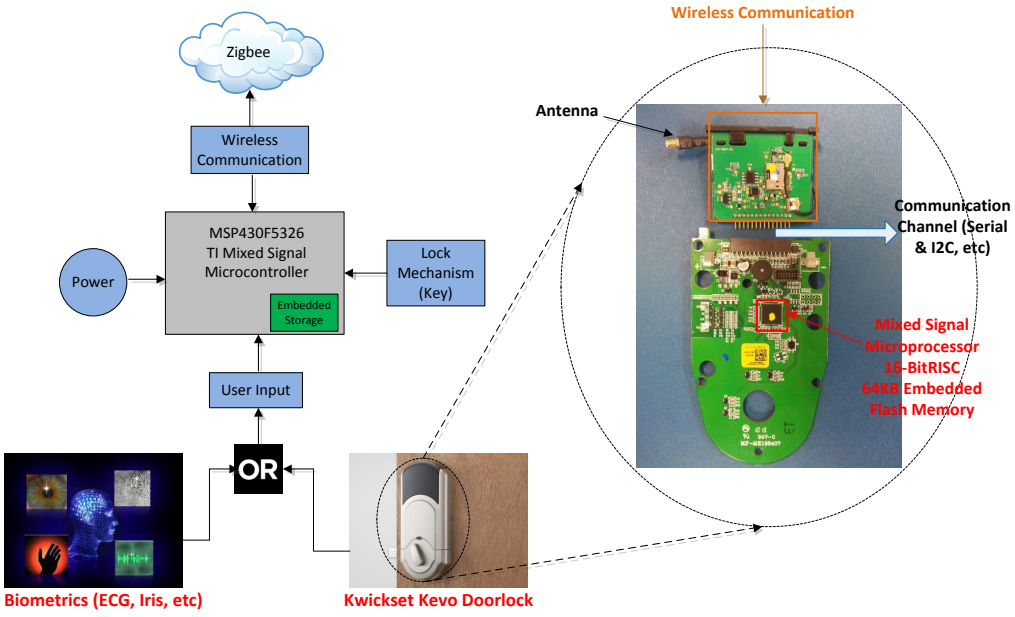


Fig. 5.5: Illustration of additional authentication methods to existing smart lock model (Kwikset Kevo).

Recent Attacks to IoT Devices

Throughout the adaptation and implementation of the IoT model over the past few years there have been a string of recent attacks on IoT device authentication [97], [25], [96], [117], [30], [11]; for example, attacks on smart door locks. In addition to this work, it has been shown that other forms of biometric authentication (e.g. fingerprint authentication) can be easily reproduced/replicated. For these reasons, this work proposes a more effective and reliable method for implementing biometric authentication using current, standard IoT devices.

5.6 Conclusion

In this chapter, we proposed a solution to the results of our investigation of the infrastructure of IoT devices in their application to authentication within the healthcare domain. Through our solution one would know that the information produced from an IoT is trustworthy, that the individuals accessing information can be properly identified, and that the exchange of sensitive biological signals across a network is secure. Furthermore, we showed that the hardware-based solutions that are presented in this chapter are low-cost and feasibly implementable on existing resource constrained IoT systems (sensors, wearables, smart devices). To lend weight to our solution, we also examined the use of a popular and growing authentication method (human characteristics based technique) to illustrate

how hardware verification can improved upon an existing framework (e.g. Smart DoorLock IoT device). The Internet has changed drastically the way we live, moving interactions between people at a virtual level in several contexts spanning from the professional life to social relationships. The IoT has the potential to add a new dimension to this process by enabling communications with and among smart objects, thus leading to the vision of anytime, anywhere, anything communications [7]. As the IoT continues to grow, we will need a strong method for authenticating hardware. Our solutions present an effective, efficient, and lightweight method for providing device verification in the complex environment of the Internet of Things.

Chapter 6

Conclusions

Security is a critical issue in modern information systems particularly hardware security which has merged as one of the most serious challenges due to electronic devices penetrating every aspect of our life. In general, hardware needs to be protected since its insecurities can facilitate attacks on the programs and contents running on it. Furthermore, manufacturing of integrated circuits (ICs) by potentially untrusted foundries using possibly untrusted CAD tools or insecure reusable components, and the need to protect the digital rights have raised the alarm for protecting hardware [64]. Therefore, one must consider cost, power consumption, performance, and reliability while designing/proposing methods for securing physical hardware objects. Most present hardware security techniques use conventional cryptographic protocols to provide security. In traditional cryptographic protocols, secrecy is provided by trapdoor mathematical functions and digital keys that make the protocols resilient to algorithmic attacks. Thus, hardware security primitives are strong solutions in order to protect hardware component.

However, with the crucial need for information and data security, cryptography has become more and more important. Cryptographic systems, in order to ensure their security guarantees, are highly dependent on the availability of a truly random numbers. Random Number Generators (RNGs) have thus become an indispensable primitive in security systems and applications. In this thesis, we have presented several novel designs and architectures for hardware-based random functions security primitives. The major contributions of the thesis will be presented in this chapter.

6.1 DRAM-based Physical Unclonable Functions

Physical Unclonable functions are a very promising method as a hardware security primitive. A PUF is an irreversible probabilistic function that produces a random bit string. It is simple to implement but hard to predict and emulate. PUFs have been widely proposed as security primitives to provide device identification and authentication. In chapter 2, we proposed a novel dynamic memory based PUF (DRAM PUF) for the authentication of electronic hardware systems. The DRAM PUF relies on the fact that the capacitor in the DRAM initializes to random values at startup time. Most PUF designs require custom circuits to convert unique analog characteristics into digital bits but with using our method, no extra circuitry is required to achieve a reliable 128 bit PUF. Our results showed that

the proposed DRAM PUF provides a large number of input patterns (challenges) when compared to other memory-based PUFs circuits such as Static RAM PUFs. Our DRAM PUFs provided highly unique PUFs with a 0.4937 average inter-die Hamming Distance. We also proposed an enrollment algorithm to achieve highly reliable results to generate PUF Identifications for system level security. This algorithm has been validated on real DRAMs with an experimental setup to test different operating conditions.

6.2 DRAM-based Random Number Generations

Due to the need for highly secured electronic information systems, almost every important and valuable document or piece of data is stored/transferred in some type of encrypted form to prevent attackers from compromising privacy or stealing information for nefarious uses. High entropy random numbers from physical sources are a critical component in authentication and encryption processes within secure systems. Secure encryption is dependent on sources of truly random numbers for generating keys, and there is a need for an on chip random number generator to achieve adequate security. A TRNG is an important security primitive used in a variety of applications including cryptographic algorithms, communication systems, simulations, etc. It is critical to be able to produce outputs consisting of fully unpredictable and unbiased bits in a cost-effective manner.

In chapter 3, we presented a robust hardware TRNG based on the Dynamic RAM (DRAM) remanence effect, which is a condition whereby information remains in a DRAM even after powering it down. The advantage of our hardware TRNG is that it forms from existing components with no extra circuitry. We assessed and tested the randomness of our proposed hardware TRNG by applying the NIST Statistical Test which indicates the unpredictability and non-repeatability of our data. Given its strong NIST results, we believe that there is a potential for immediate cryptographic applications.

6.3 Power Supply Noise-based True Random Number Generations

Random numbers are used in virtually every form of encryption or data security applications. TRNGs derive their randomness from a physical entropy source and have been developed partly to enhance the security of PRNGs by providing seeds that are inherently non-deterministic. TRNGs are central to many computing applications, particularly in security domains such as cryptography. In chapter 4, we considered the design and implementation of a low-cost and lightweight TRNG. In the interest of being thorough, we examined six different power supplies in order to verify the non-cyclostationary behavior of the voltage sources. Our novel TRNG model is based on power supply variations (noise behavior) and self-adjusting operation. The benefits of this novel design are that: it is simple and

easy to implement, there is little to no additional cost required to incorporate the TRNG into existing circuitry, and the addition of Dynamic Voltage Feedback Tuning (which we call DVFT) allows for feasibility and robustness of our model. The cumulative affect of these benefits is the practicality of the entire power-supply noise based TRNG system. We then validate the results of our theoretical model and experimental setup to show that there is a high entropy rate based on the findings from the NIST Statistical Test Suite. Based on our observations and results, our DVFT power-supply noise based TRNG model has the potential to be used in critical applications while also having the advantage of simplicity and practicality.

6.4 Hardware Security Architectures for the Internet of Things

The Internet of Things (IoT) is a design implementation of embedded system design that connects a variety of devices, sensors, and physical objects to a larger connected network (e.g. the Internet) which requires human-to-human or human-to-computer interaction. While the IoT is expected to expand the users connectivity and everyday convenience, there are serious security considerations that come into account when using the IoT for distributed authentication. Chapter 5 examined the application of the Internet of Things within two case studies in the healthcare domain and proposed architecture for existing IoT smart devices. In

Case Study 1, our aim was to investigate the issues with the current method of device and network authentication of IoT peripherals within healthcare. Therefore, we proposed a low-cost solution to design concerns for the application of IoT devices for authentication (device verification and patient authentication. In Case Study 2, we examined the incorporation of human characteristic-based authentication to IoT design in order to evaluate the possibility of feasibility and cost of this implementation in an existing IoT device (Smart DoorLock IoT device).

6.5 Future Work

While this thesis has demonstrated the potential of efficiently designing hardware-based random functions, many opportunities for extending the scope of this thesis remain. However, The research presented in this thesis seems to have raised more questions that it has answered. There are several lines of research arising from this work which should be pursued in near future. This section presents some of these directions. The following ideas could be tested:

Firstly, we intend to investigate DRAM remanence on different memory platforms. We have plan to consider different operating conditions (aging, temperature and voltage) and their impact on our results. Different environmental conditions have effect of the remanence time of different memory platform. We eager to investigate these effects on the randomness of the bit strings.

A second line of research, which follows from Chapter 5, is to implement our authentication design models for the Internet of Things and embedded systems.

Finally, we have plan to work on investigating possible attacks on wearable devices using Machine Learning (ML) techniques in order to help us find the countermeasures.

6.6 Summary of Conclusions

This thesis is devoted to developing a series of designs and architectures based on hardware random function to tackle the issues and vulnerabilities to hardware objects, in order to protect them from malicious attacks, counterfeiting, reverse engineering, etc. Our hardware-based random functions security primitives provide low-cost, lightweight, efficient, and secure hardware platforms for the embedded systems.

Bibliography

[1] T. Adegbija, A. Rogacs, C. Patel, and A. Gordon-Ross, *Microprocessor optimizations for the Internet of things: A survey*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **PP** (2017), no. 99, 1–1.

[2] Takehiko Amaki, Masanori Hashimoto, and Takao Onoye, *An oscillator-based true random number generator with jitter amplifier*, Circuits and Systems (ISCAS), 2011 IEEE International Symposium on, IEEE, 2011, pp. 725–728.

[3] ———, *An oscillator-based true random number generator with process and temperature tolerance*, Design Automation Conference (ASP-DAC), 2015 20th Asia and South Pacific, IEEE, 2015, pp. 4–5.

[4] Scott Amyx, *Privacy dangers of wearables and the Internet of things*, Managing Security Issues and the Hidden Dangers of Wearable Technologies (2016), 131–142.

[5] Orlando Arias, Kelvin Ly, and Yier Jin, *Security and privacy in IoT era*, Smart Sensors at the IoT Frontier, Springer, 2017, pp. 351–378.

[6] Sakthivel Arunprasanth, Udaya D Annakkage, Chandana Karawita, and Rick Kuffel, *Generalized frequency-domain controller tuning procedure for VSC systems*, IEEE Transactions on Power Delivery **31** (2016), no. 2, 732–742.

[7] Luigi Atzori, Antonio Iera, and Giacomo Morabito, *The Internet of things: A survey*, Computer networks **54** (2010), no. 15, 2787–2805.

[8] Francesco Beritelli and Salvatore Serrano, *Biometric identification based on frequency analysis of cardiac sounds*, IEEE Transactions on Information Forensics and Security **2** (2007), no. 3, 596–604.

[9] Mudit Bhargava, Kaship Sheikh, and Ken Mai, *Robust true random number generator using hot-carrier injection balanced metastable sense amplifiers*,

Hardware Oriented Security and Trust (HOST), 2015 IEEE International Symposium on, IEEE, 2015, pp. 7–13.

[10] Lena Biel, Ola Pettersson, Lennart Philipson, and Peter Wide, *ECG analysis: a new approach in human identification*, IEEE Transactions on Instrumentation and Measurement **50** (2001), no. 3, 808–812.

[11] Jon Brodkin, *Comcast security flaw could help burglars break into homes undetected*, `http://arstechnica.com/security/2016/01/comcast-security/`, 2016.

[12] Helena Bruyninckx, Fredaic Lafitte, and Dirk Van Heule, *Safe cryptographic random number generation using untrusted generators*, Communications (ICC), 2014 IEEE International Conference on, IEEE, 2014, pp. 731–736.

[13] Wayne Burleson and Salma Mirza, *Analysis of on-chip true random number generators based on power supply variation*, RFID Security Workshop, 2008.

[14] M Cavalleri, R Morstabilini, and G Reni, *A wearable device for a fully automated in-hospital staff and patient identification*, Engineering in Medicine and Biology Society, 2004. IEMBS'04. 26th Annual International Conference of the IEEE, vol. 2, IEEE, 2004, pp. 3278–3281.

[15] Jose Juan Mijares Chan, Bhanu Sharma, Jiaqing Lv, Gabriel Thomas, Ruppa Thulasiram, and Parimala Thulasiraman, *True random number generator using GPUs and histogram equalization techniques*, High Performance Computing and Communications (HPCC), 2011 IEEE 13th International Conference on, IEEE, 2011, pp. 161–170.

[16] Wenjie Che, Fareena Saqib, and Jim Plusquellic, *PUF-based authentication*, Computer-Aided Design (ICCAD), 2015 IEEE/ACM International Conference on, IEEE, 2015, pp. 337–344.

[17] Xiaoming Chen, Lin Wang, Boxun Li, Yu Wang, Xin Li, Yongpan Liu, and Huazhong Yang, *Modeling random telegraph noise as a randomness source and its application in true random number generation*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **35** (2016), no. 9, 1435–1448.

[18] Kim-Kwang Raymond Choo, Mehran Mozaffari Kermani, Reza Azarderakhsh, and Manimaran Govindarasu, *Emerging embedded and cyber physical system security challenges and innovations*, IEEE Transactions on Dependable and Secure Computing **14** (2017), no. 3, 235–236.

[19] Lawrence T Clark, James Adams, and Keith E Holbert, *Secure true random number generation using 1.5-T transistor flash memory*, September 26 2016, US Patent App. 15/276,087.

[20] GB Clarke, D Van Dijk, and SM Devadas, *Controlled physical random functions*, Proceedings. 18th Annual (2002), 149–160.

[21] Jayita Das, Kevin Scott, Srinath Rajaram, Drew Burgett, and Sanjukta Bhanja, *MRAM PUF: A novel geometry based magnetic PUF with integrated CMOS*, IEEE Transactions on Nanotechnology **14** (2015), no. 3, 436–443.

[22] Norbert Deák, Tamás Györfi, Kinga Márton, Lucia Vacariu, and Octavian Cret, *Highly efficient true random number generator in FPGA devices using phase-locked loops*, Control Systems and Computer Science (CSCS), 2015 20th International Conference on, IEEE, 2015, pp. 453–458.

[23] Kresimir Delac and Mislav Grgic, *A survey of biometric recognition methods*, Electronics in Marine, 2004. Proceedings Elmar 2004. 46th International Symposium, IEEE, 2004, pp. 184–193.

[24] Siva Nishok Dhanuskodi, Arunkumar Vijayakumar, and Sandip Kundu, *A chaotic ring oscillator based random number generator*, Hardware-Oriented Security and Trust (HOST), 2014 IEEE International Symposium on, IEEE, 2014, pp. 160–165.

[25] Paul Ducklin, *IoT security in the spotlight at PrivacyCon*, `https://nakedsecurity.sophos.com/2016/01/22/iot-security-in-the-spotlight-at-privacycon/`, 2016.

[26] Patrick R Gallagher, *A guide to understanding data remanence in automated information systems*, 1991.

[27] Dinesh Ganta and Leyla Nazhandali, *Study of IC aging on ring oscillator physical unclonable functions*, Quality Electronic Design (ISQED), 2014 15th International Symposium on, IEEE, 2014, pp. 461–466.

[28] Achiranshu Garg and Tony T Kim, *Design of SRAM PUF with improved uniformity and reliability utilizing device aging effect*, Circuits and Systems (ISCAS), 2014 IEEE International Symposium on, IEEE, 2014, pp. 1941–1944.

[29] Blaise Gassend, Dwaine Clarke, Marten Van Dijk, and Srinivas Devadas, *Silicon physical random functions*, Proceedings of the 9th ACM conference on Computer and communications security, ACM, 2002, pp. 148–160.

[30] Dan Goodin, *Why Algebraic Eraser may be the riskiest cryptosystem you've never heard of*, `http://arstechnica.com/security/2015/11/why-algebraiceraser-maybe-the-most-risky-cryptosystem/`, 2015.

[31] Michael Gruhn and Tilo Muller, *On the practicability of cold boot attacks*, Availability, Reliability and Security (ARES), 2013 Eighth International Conference on, IEEE, 2013, pp. 390–397.

[32] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami, *Internet of things (IoT): A vision, architectural elements, and future directions*, Future generation computer systems **29** (2013), no. 7, 1645–1660.

[33] Tim Güneysu, *True random number generation in block memories of reconfigurable devices*, Field-Programmable Technology (FPT), 2010 International Conference on, IEEE, 2010, pp. 200–207.

[34] Peter Gutmann, *Secure deletion of data from magnetic and solid-state memory*, Proceedings of the Sixth USENIX Security Symposium, San Jose, CA, vol. 14, 1996, pp. 77–89.

[35] ———, *Data remanence in semiconductor devices*, Proceedings of the 10th conference on USENIX Security Symposium-Volume 10, USENIX Association, 2001, p. 4.

[36] J Alex Halderman, Seth D Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A Calandrino, Ariel J Feldman, Jacob Appelbaum, and Edward W Felten, *Lest we remember: cold-boot attacks on encryption keys*, Communications of the ACM **52** (2009), no. 5, 91–98.

[37] Maryam S Hashemian, Bhanu Singh, Francis Wolff, Daniel Weyer, Steve Clay, and Christos Papachristou, *A robust authentication methodology using physically unclonable functions in DRAM arrays*, Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition, EDA Consortium, 2015, pp. 647–652.

[38] Yuhao He, Weijun Zhang, Hui Zhou, Lixing You, Chaolin Lv, Lu Zhang, Xiaoyu Liu, Junjie Wu, Sijing Chen, Min Ren, et al., *Bias-free true random number generation using superconducting nanowire single-photon detectors*, Supercond. Sci. Technol **29** (2016), no. 085005, 085005.

[39] N Henze and B Zirkler, *A class of invariant consistent tests for multivariate normality*, Communications in Statistics-Theory and Methods **19** (1990), no. 10, 3595–3617.

[40] Charles Herder, Meng-Day Yu, Farinaz Koushanfar, and Srinivas Devadas, *Physical unclonable functions and applications: A tutorial*, Proceedings of the IEEE **102** (2014), no. 8, 1126–1141.

[41] Daniel E Holcomb, Wayne P Burleson, and Kevin Fu, *Power-up SRAM state as an identifying fingerprint and source of true random numbers*, IEEE Transactions on Computers **58** (2009), no. 9, 1198–1210.

[42] Alison Hosey, Md Tauhidur Rahman, Kan Xiao, Domenic Forte, and Mohammad Tehranipoor, *Advanced analysis of cell stability for reliable sram pufs*, Test Symposium (ATS), 2014 IEEE 23rd Asian, IEEE, 2014, pp. 348–353.

[43] Anirudh Iyengar, Kenneth Ramclam, and Swaroop Ghosh, *Dwm-puf: A low-overhead, memory-based security primitive*, Hardware-Oriented Security and Trust (HOST), 2014 IEEE International Symposium on, IEEE, 2014, pp. 154–159.

[44] Benjamin Jun and Paul Kocher, *The Intel random number generator*, Cryptography Research Inc. white paper (1999).

[45] Sung-Mo Kang and Yusuf Leblebici, *CMOS digital integrated circuits*, Tata McGraw-Hill Education, 2003.

[46] Naghmeh Karimi, Jean-Luc Danger, Florent Lozach, and Sylvain Guilley, *Predictive aging of reliability of two delay PUFs*, SPACE, 2016, pp. 213–232.

[47] Nima Karimian, Zimu Guo, Mark Tehranipoor, and Domenic Forte, *Highly reliable key generation from electrocardiogram (ECG)*, IEEE Transactions on Biomedical Engineering **64** (2017), no. 6, 1400–1411.

[48] ———, *Human recognition from photoplethysmography (ppg) based on non-fiducial features*, Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on, IEEE, 2017, pp. 4636–4640.

[49] Nima Karimian, Fatemeh Tehranipoor, Md Tauhidur Rahman, Shane Kelly, and Domenic Forte, *Genetic algorithm for hardware trojan detection with ring oscillator network (RON)*, Technologies for Homeland Security (HST), 2015 IEEE International Symposium on, IEEE, 2015, pp. 1–6.

[50] Nima Karimian, Mark Tehranipoor, and Domenic Forte, *Non-fiducial ppg-based authentication for healthcare application*, Biomedical & Health Informatics (BHI), 2017 IEEE EMBS International Conference on, IEEE, 2017, pp. 429–432.

[51] Nima Karimian, Paul A Wortman, and Fatemeh Tehranipoor, *Evolving authentication design considerations for the internet of biometric things (IoBT)*, Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2016 International Conference on, IEEE, 2016, pp. 1–10.

[52] Christoph Keller, Frank Gurkaynak, Hubert Kaeslin, and Norbert Felber, *Dynamic memory-based physically unclonable function for the generation of unique identifiers and true random numbers*, Circuits and Systems (ISCAS), 2014 IEEE International Symposium on, IEEE, 2014, pp. 2740–2743.

[53] Patrick Koeberl, Ünal Kocabaş, and Ahmad-Reza Sadeghi, *Memristor PUFs: a new generation of memory-based physically unclonable functions*, Proceedings of the Conference on Design, Automation and Test in Europe, EDA Consortium, 2013, pp. 428–431.

[54] Robert Konig, Renato Renner, and Christian Schaffner, *The operational meaning of min-and max-entropy*, IEEE Transactions on Information theory **55** (2009), no. 9, 4337–4347.

[55] Siew-Hwee Kwok, Yen-Ling Ee, Guanhan Chew, Kanghong Zheng, Khoongming Khoo, and Chik-How Tan, *A comparison of post-processing techniques for biased random number generators*, IFIP International Workshop on Information Security Theory and Practices, Springer, 2011, pp. 175–190.

[56] Dongsheng Liu, Zilong Liu, Lun Li, and Xuecheng Zou, *A low-cost low-power ring oscillator-based truly random number generator for encryption on smart cards*, IEEE Transactions on Circuits and Systems II: Express Briefs **63** (2016), no. 6, 608–612.

[57] Wenchao Liu, Zhenhua Zhang, Miaoxin Li, and Zhenglin Liu, *A trustworthy key generation prototype based on DDR3 PUF for wireless sensor networks*, Sensors **14** (2014), no. 7, 11542–11556.

[58] Ingo Ltkebohle, *Gartner's hype cycle special report for 2011*, `http://www.gartner.com/technology/research/hype-cycles`, 2011.

[59] Roel Maes and Vincent van der Leest, *Countering the effects of silicon aging on SRAM PUFs*, Hardware-Oriented Security and Trust (HOST), 2014 IEEE International Symposium on, IEEE, 2014, pp. 148–153.

[60] Abhranil Maiti, Jeff Casarona, Luke McHale, and Patrick Schaumont, *A large scale characterization of RO-PUF*, Hardware-Oriented Security and Trust (HOST), 2010 IEEE International Symposium on, IEEE, 2010, pp. 94–99.

[61] Abhranil Maiti, Inyoung Kim, and Patrick Schaumont, *A robust physical unclonable function with enhanced challenge-response set*, IEEE Transactions on Information Forensics and Security **7** (2012), no. 1, 333–345.

[62] Abhranil Maiti, Logan McDougall, and Patrick Schaumont, *The impact of aging on an FPGA-based physical unclonable function*, Field Programmable Logic and Applications (FPL), 2011 International Conference on, IEEE, 2011, pp. 151–156.

[63] Abhranil Maiti and Patrick Schaumont, *The impact of aging on a physical unclonable function*, IEEE Transactions on Very Large Scale Integration (VLSI) Systems **22** (2014), no. 9, 1854–1864.

[64] Mehrdad Majzoobi, Farinaz Koushanfar, and Miodrag Potkonjak, *Testing techniques for hardware security*, Test Conference, 2008. ITC 2008. IEEE International, IEEE, 2008, pp. 1–10.

[65] Andrei Marghescu, Daniel-Ciprian Vasile, Paul Svasta, and Emil Simion, *Personalized ring oscillator-based true random number generator analysis using non-invasive attacks*, Design and Technology in Electronic Packaging (SIITME), 2016 IEEE 22nd International Symposium for, IEEE, 2016, pp. 98–101.

[66] A Markettos and Simon Moore, *The frequency injection attack on ring-oscillator-based true random number generators*, Cryptographic Hardware and Embedded Systems-CHES 2009 (2009), 317–331.

[67] Honorio Martin, Thomas Korak, Enrique San Millán, and Michael Hutter, *Fault attacks on strngs: Impact of glitches, temperature, and underpowering on randomness*, IEEE transactions on information forensics and security **10** (2015), no. 2, 266–277.

[68] Anas Mazady, Md Tauhidur Rahman, Domenic Forte, and Mehdi Anwar, *Memristor pufa security primitive: Theory and experiment*, IEEE Journal on Emerging and Selected Topics in Circuits and Systems **5** (2015), no. 2, 222–229.

[69] Sergey Morozov, Abhranil Maiti, and Patrick Schaumont, *An analysis of delay based PUF implementations on FPGA*, ARC, Springer, 2010, pp. 382–387.

[70] Cícero Nunes, Paulo F Butzen, André I Reis, and Renato P Ribas, *BTI, HCI and TDDB aging impact in flip–flops*, Microelectronics Reliability **53** (2013), no. 9, 1355–1359.

[71] Ikenna Odinaka, Po-Hsiang Lai, Alan D Kaplan, Joseph A O'Sullivan, Erik J Sirevaag, and John W Rohrbaugh, *ECG biometric recognition: A comparative analysis*, IEEE Transactions on Information Forensics and Security **7** (2012), no. 6, 1812–1824.

[72] David W Osten, Hatim M Carim, Michael R Arneson, and Bradford L Blan, *Biometric, personal authentication system*, February 17 1998, US Patent 5,719,950.

[73] Erdinc Ozturk, Ghaith Hammouri, and Berk Sunar, *Physical unclonable function with tristate buffers*, Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on, IEEE, 2008, pp. 3194–3197.

[74] Shashi Kant Pandey, Soumya R Mohanty, and Nand Kishor, *A literature survey on load–frequency control for conventional and distribution generation power systems*, Renewable and Sustainable Energy Reviews **25** (2013), 318–334.

[75] Craig S Petrie and J Alvin Connelly, *A noise-based IC random number generator for applications in cryptography*, IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications **47** (2000), no. 5, 615–621.

[76] Miodrag Potkonjak and Vishwa Goudar, *Public physical unclonable functions*, Proceedings of the IEEE **102** (2014), no. 8, 1142–1156.

[77] Pravin Prabhu, Ameen Akel, Laura M Grupp, S Yu Wing-Kei, G Edward Suh, Edwin Kan, and Steven Swanson, *Extracting device fingerprints from flash memory by exploiting physical variations*, International Conference on Trust and Trustworthy Computing, Springer, 2011, pp. 188–201.

[78] M. T. Rahman, F. Rahman, D. Forte, and M. Tehranipoor, *An aging-resistant ro-puf for reliable key generation*, IEEE Transactions on Emerging Topics in Computing **4** (2016), no. 3, 335–348.

[79] M Tauhidur Rahman, Domenic Forte, Xiaoxiao Wang, and Mark Tehranipoor, *Enhancing noise sensitivity of embedded srams for robust true random number generation in SoCs*, Hardware-Oriented Security and Trust (AsianHOST), IEEE Asian, IEEE, 2016, pp. 1–6.

[80] Md Tauhidur Rahman, Domenic Forte, Fahim Rahman, and Mark Tehranipoor, *A pair selection algorithm for robust ro-puf against environmental variations and aging*, Computer Design (ICCD), 2015 33rd IEEE International Conference on, IEEE, 2015, pp. 415–418.

[81] Md Tauhidur Rahman, Kan Xiao, Domenic Forte, Xuhei Zhang, Jerry Shi, and Mohammad Tehranipoor, *TI-TRNG: Technology independent true random number generator*, Proceedings of the 51st Annual Design Automation Conference, ACM, 2014, pp. 1–6.

[82] Tauhidur Rahman, Domenic Forte, Jim Fahrny, and Mohammad Tehranipoor, *Aro-puf: An aging-resistant ring oscillator puf design*, Proceedings of the conference on Design, Automation & Test in Europe, European Design and Automation Association, 2014, p. 69.

[83] Amir Rahmati, Mastooreh Salajegheh, Dan Holcomb, Jacob Sorber, Wayne P Burleson, and Kevin Fu, *TARDIS: Time and remanence decay in sram to implement secure protocols on embedded devices without clocks*, Proceedings of the 21st USENIX conference on Security symposium, USENIX Association, 2012, pp. 36–36.

[84] Milana Ram, *Reliability analysis of dynamic logic circuits under transistor aging effects in nanotechnology*, Ph.D. thesis, San Francisco State University, 2010.

[85] Mandeep Singh Randhawa, *Analysis of impact of transistor aging effects on clock skew in nano-scale CMOS*, Ph.D. thesis, San Francisco State University, California, 2011.

[86] Garrett S Rose, Nathan McDonald, Lok-Kwong Yan, Bryant Wysocki, and Karen Xu, *Foundations of memristor based PUF architectures*, Nanoscale Architectures (NANOARCH), 2013 IEEE/ACM International Symposium on, IEEE, 2013, pp. 52–57.

[87] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, and Elaine Barker, *A statistical test suite for random and pseudorandom number generators for cryptographic applications*, Tech. report, Booz-Allen and Hamilton Inc Mclean Va, 2001.

[88] Min-Woo Ryu, Jaeho Kim, Sang-Shin Lee, and Min-Hwan Song, *Survey on Internet of things*, SmartCR **2** (2012), no. 3, 195–202.

[89] WJ Sarjeant, FW MacDougall, DW Larson, and I Kohlberg, *Energy storage capacitors: aging, and diagnostic approaches for life validation*, IEEE Transactions on Magnetics **33** (1997), no. 1, 501–506.

[90] André Schaller, Wenjie Xiong, Nikolaos Athanasios Anagnostopoulos, Muhammad Umair Saleem, Sebastian Gabmeyer, Stefan Katzenbeisser, and Jakub Szefer, *Intrinsic rowhammer PUFs: Leveraging the rowhammer effect*

for improved security, Hardware Oriented Security and Trust (HOST), 2017 IEEE International Symposium on, IEEE, 2017, pp. 1–7.

[91] Geert-Jan Schrijen and Vincent van der Leest, *Comparative analysis of SRAM memories used as PUF primitives*, Proceedings of the Conference on Design, Automation and Test in Europe, EDA Consortium, 2012, pp. 1319–1324.

[92] Mario Stipčević and Çetin Kaya Koç, *True random number generators*, Open Problems in Mathematics and Computational Science, Springer, 2014, pp. 275–315.

[93] Fahim Sufi, Ibrahim Khalil, and Jiankun Hu, *ECG-based authentication*, Handbook of information and communication security (2010), 309–331.

[94] G Edward Suh and Srinivas Devadas, *Physical unclonable functions for device authentication and secret key generation*, Proceedings of the 44th annual design automation conference, ACM, 2007, pp. 9–14.

[95] Soubhagya Sutar, Arnab Raha, and Vijay Raghunathan, *D-PUF: An intrinsically reconfigurable DRAM PUF for device authentication in embedded systems*, Compliers, Architectures, and Sythesis of Embedded Systems (CASES), 2016 International Conference on, IEEE, 2016, pp. 1–10.

[96] Lisa Vaas, *IoT doorbell have up Wi-Fi passwords to anybody with a screwdriver*, `https://nakedsecurity.sophos.com/2016/01/27/iot-doorbell-gave-up-wi-fi-passwords-to-anybody-with\-a-screwdriver/`, 2016.

[97] _____, *We might use your IoT stuff to spy on you, says top spook James Clapper*, `https://nakedsecurity.sophos.com/2016/02/11/we-might-use-your-iot-stuff-to-spy-on-you-says-top-spook\-james-clapper`, 2016.

[98] Elena Ioana Vatajelu, Giorgio Di Natale, Marco Indaco, and Paolo Prinetto, *STT MRAM-based PUFs*, Design, Automation & Test in Europe Conference & Exhibition (DATE), 2015, IEEE, 2015, pp. 872–875.

[99] AM Vilamovska, E Hattziandreu, R Schindler, C Van Oranje, H De Vries, and J Krapelse, *RFID application in healthcare–scoping and identifying areas for RFID deployment in healthcare delivery*, RAND Europe, February (2009).

[100] Xiaoxiao Wang and Mohammad Tehranipoor, *Novel physical unclonable function with process and environmental variations*, Proceedings of the Conference on Design, Automation and Test in Europe, European Design and Automation Association, 2010, pp. 1065–1070.

[101] Xiaoxiao Wang, LeRoy Winemberg, Donglin Su, Dat Tran, Saji George, Nisar Ahmed, Steve Palosh, Allan Dobin, and Mohammad Tehranipoor, *Aging adaption in integrated circuits using a novel built-in sensor*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **34** (2015), no. 1, 109–121.

[102] Yinglei Wang, Wing-kei Yu, Shuo Wu, Greg Malysa, G Edward Suh, and Edwin C Kan, *Flash memory for ubiquitous hardware security functions: True random number generation and device fingerprints*, Security and Privacy (SP), 2012 IEEE Symposium on, IEEE, 2012, pp. 33–47.

[103] J West, T Kohno, D Lindsay, and J Sechman, *Wearfit: Security design analysis of a wearable fitness tracker*, Technical report, IEEE Center for Secure Design (2016).

[104] Piotr Zbigniew Wieczorek, *An FPGA implementation of the resolve time-based true random number generator with quality control*, IEEE Transactions on Circuits and Systems I: Regular Papers **61** (2014), no. 12, 3450–3459.

[105] Piotr Zbigniew Wieczorek and Krzysztof Golofit, *Dual-metastability time-competitive true random number generator*, IEEE Transactions on Circuits and Systems I: Regular Papers **61** (2014), no. 1, 134–145.

[106] Kan Xiao, Md Tauhidur Rahman, Domenic Forte, Yu Huang, Mei Su, and Mohammad Tehranipoor, *Bit selection algorithm suitable for high-volume production of SRAM-PUF*, Hardware-Oriented Security and Trust (HOST), 2014 IEEE International Symposium on, IEEE, 2014, pp. 101–106.

[107] Wenjie Xiong, André Schaller, Nikolaos Anagnostopoulos, Muhammad Umair Saleem, Sebastian Gabmeyer, Stefan Katzenbeisser, and Jakub Szefer, *Run-time accessible DRAM PUFs in commodity devices.*, CHES, 2016, pp. 432–453.

[108] W. Yan, C. Jin, and F. Tehranipoor andJ. A. Chandy, *Phase calibrated ring oscillator PUF design and implementation on FPGAs*, 2017 27th International Conference on Field Programmable Logic and Applications (FPL), Sep 2017.

[109] Wei Yan, Fatemeh Tehranipoor, and John A Chandy, *A novel way to authenticate untrusted integrated circuits*, Proceedings of the IEEE/ACM International Conference on Computer-Aided Design, IEEE Press, 2015, pp. 132–138.

[110] ———, *PUF-based fuzzy authentication without error correcting codes*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (2016).

[111] Venkata P Yanambaka, Saraju P Mohanty, Elias Kougianos, and Jawar Singh, *Secure multi-key generation using ring oscillator based physical unclonable function*, Nanoelectronic and Information Systems (iNIS), 2016 IEEE International Symposium on, IEEE, 2016, pp. 200–205.

[112] Kaiyuan Yang, David Blaauw, and Dennis Sylvester, *An all-digital edge racing true random number generator robust against PVT variations*, IEEE Journal of Solid-State Circuits **51** (2016), no. 4, 1022–1031.

[113] Yida Yao, MyungBo Kim, Jianmin Li, Igor L Markov, and Farinaz Koushanfar, *ClockPUF: Physical unclonable functions based on clock networks*, Proceedings of the Conference on Design, Automation and Test in Europe, EDA Consortium, 2013, pp. 422–427.

[114] Chi-En Yin, Gang Qu, and Qiang Zhou, *Design and implementation of a group-based RO PUF*, Proceedings of the Conference on Design, Automation and Test in Europe, EDA Consortium, 2013, pp. 416–421.

[115] Xuping Zhang, Li Qi, Zhiqiang Tang, and Yixin Zhang, *Portable true random number generator for personal encryption application based on smartphone camera*, Electronics Letters **50** (2014), no. 24, 1841–1843.

[116] Yu Zheng, Fengchao Zhang, and Swarup Bhunia, *Dscanpuf: A delay-based physical unclonable function built into scan chain*, IEEE Transactions on Very Large Scale Integration (VLSI) Systems **24** (2016), no. 3, 1059–1070.

[117] Zeljka Zorz, *WiFi jamming attacks more simple and cheaper than ever*, `https://www.helpnetsecurity.com/2015/10/13/wifi-jamming-attacks-more-simple-and-cheaper-than-ever/`, 2015.

Appendix A

Related Publications

1. Tehranipoor, Fatemeh, et al., "DRAM based Intrinsic Unclonable Functions for System Level Security and Physical Authentication," IEEE Transactions on Very Large Scale Integration Systems (TVLSI), 2016.

2. Tehranipoor, Fatemeh, et al., "DRAM based intrinsic physical unclonable functions for system level security," Proceedings of the 25th edition on Great Lakes Symposium on VLSI. ACM, 2015.

3. Tehranipoor, Fatemeh, et al., "Robust Hardware True Random Number Generators using DRAM Remanence Effects," IEEE Int. Symp. on Hardware-Oriented Security and Trust (HOST), 2016.

4. Tehranipoor, Fatemeh, et al., "A study of Power Supply Variation as a Source of Random Noise," 30th International Conference on VLSI Design & 16th International Conference on Embedded Systems (VLSID), 2017.

5. Tehranipoor, Fatemeh, et al., "DRAM PUFs Reliability Analysis due to Device Accelerated Aging," IEEE International Symposium on Circuits and Sys-

tems(ISCAS), 2017.

6. Karimian, Nima, et al., "Noise Assessment Framework for Optimizing ECG-based Key Generation," IEEE International Symposium on Technologies for Homeland Security (HST), 2017.

7. Wortman, P., Tehranipoor, F., Karimian, N., and Chandy, J., "Proposing a Modeling Framework for Preventing Resource Constraints and Over-Engineering in Healthcare Domain," IEEE-EMBS International Conference On Biomedical And Health Informatics (BHI), 2017.

8. Yan, Wei, et al., "PUF-based Fuzzy Authentication without Error Correcting Codes," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), 2016.

9. Yan, Wei, et al., "Phase Calibration PUF Design and Implementation on FPGA," 27th International Conference on Field-Programmable Logic and Applications (FPL), 2017.

10. Karimian, Nima, et al., "Evolving Authentication Design Considerations of the Internet of Biometric Things (IoBT)," Proceedings of the Eleventh IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES), 2016.

11. Yan, Wei, et al., "A Novel Way to Authenticate Untrusted Integrated Circuits," In 2015 IEEE International Conference On Computer Aided Design (IC-

CAD), 2015.

12. Karimian, Nima, et al., "Genetic Algorithm for Hardware Trojan Detection with Ring Oscillator Network (RON)," IEEE International Conference on Technologies for Homeland Security (HST), 2015.

13. Eckert, Charlie, et al., "DRNG: DRAM-based Random Number Generation using its Startup Value Behavior," 60th IEEE International Midwest Symposium on Circuits and Systems (MWSCAS), 2017.

14. Okwuosah, Somtochukwu, et al., "Multi-Communication Type Debugging Probe," IEEE MIT Undergraduate Research Technology Conference, 2016.