Reliable, Secure, Efficient Physical Unclonable Functions

Submitted in partial fulfillment of the requirements for

the degree of

Doctor of Philosophy

in

Electrical and Computer Engineering

Mudit Bhargava

B.Tech., Dept. of E&ECE, Indian Institute of Technology, Kharagpur
M.S., Dept. of ECE, Carnegie Mellon University

Carnegie Mellon University
Pittsburgh, PA

May, 2013

Abstract

A Physical Unclonable Function (PUF) is a die specific random function that can be used in a number of secure IC applications including die identification/authentication and key generation. At the core of a silicon PUF is a circuit (the PUF core) that generates random bits. These bits are like a silicon biometric, unique across dies, but can be reliably reproduced multiple times on a die across voltage and temperature variations and aging. In this thesis, we discuss various aspects of a PUF design, with a strong focus on circuit details. We describe the properties and the figures of merit of a PUF and then compare various PUF core implementations. We identify the biggest design challenges and provide several solutions. Results in this work are supported by measurement data from testchips designed in 65nm bulk CMOS.

We provide an apples-to-apples comparison of various PUF implementations. We demonstrate that while adequate randomness and uniqueness is achievable for most PUF implementations, achieving high reliability is a challenge. The conventional method to achieve high reliability is to use error correcting codes (ECC). Unfortunately, the overheads associated with these techniques grows very quickly with error correction capability. Alternately, researchers have proposed several error *reduction* techniques to minimize the use of error *correction*. In this work, we evaluate four orthogonal and complementary error reduction techniques. Two of these techniques, categorized as *extrinsic* techniques, are multiple evaluation (ME) and activation control (AC) and are able to reduce errors in PUF response by $\sim$70-80%. The other two techniques, categorized as *intrinsic*, are post-silicon selection (PSS) and directed accelerated aging (DAA) and show $\sim$100% correction of errors. As a proof of concept prototype, we describe a self-contained, BIST-controlled key-generator that implements the PSS technique to autonomously generate bits with a bit error rate $< 5 * 10^{-9}$, equivalent to a 128-bit key error rate $< 10^{-6}$. We also describe a realization of a strong-PUF that uses these highly reliable bits in conjunction with an Advanced Encryption Standard (AES) primitive.

Acknowledgments

I would first like to thank my advisor Prof. Ken Mai for his invaluable guidance and encouragement throughout my graduate career. I am grateful to him for providing me with the opportunities, for trusting me to explore new research areas, and for his constant and easy availability to discuss ideas and progress.

I thank Prof. Srini Devadas, Dr. Pim Tuyls, Prof. Shawn Blanton, and Dr. Sanu Mathew for taking time out of their schedule to serve on my defense committee. Their insightful feedback and words of encouragement have helped in making this work more comprehensive. I also acknowledge the funding from CyLab, the Center for Circuits & Systems Solutions (C2S2), and National Science Foundation (NSF) that has supported this research work over the years.

I had the privilege of knowing and working with some wonderful colleagues in my research group and many of them have contributed towards this work. I have spent countless hours discussing research ideas, and everything else under the sun, with Mark McCartney. His contributions, along with those of Cagla Cakir, my most frequent student co-author, in my research work have been invaluable. I also thank Craig Teegarden, Burak Erbagci, Yun Du, and Weinan Ma for their contributions during the tapeouts.

My cricket teammates - the Gladiators, the Titans, the Kents, and the Tartans - have over the years helped me maintain a balanced life. I fondly remember the time spent with my fellow members at Indian Graduate Students Association (IGSA) in organizing various events on campus and during the dramatics productions. I thank my “bakar” and “rpcfw” friends who have made the years more enjoyable.

I am grateful to my parents for the values they have taught me over the years. I thank them and my parents-in-law for their encouragement and for being supportive of my decision to pursue graduate studies. I have been fortunate to have my brother Arpit and my sister-in-law Astha around over the years and they have always helped me enjoy the lighter side of life. And my one year old son Ansh, who has helped in his own charming ways to reduce the stress in the last year of my graduate career. Above all, I thank my wonderful wife Ayesha for all her sacrifices, support, and encouragement. I cannot imagine how I could have completed this work without the serenity and sanity that her presence provides. I humbly dedicate this work to her.

Contents

1 Introduction **1**

1.1 What is a PUF? 1

1.1.1 Challenge and Response 1

1.1.2 Silicon PUFs and Non-Silicon PUFs 2

1.1.3 The PUF Core 2

1.2 Physical Unclonable Functions: Metrics 3

1.2.1 Security Metric: Uniqueness 3

1.2.2 Security Metric: Randomness 4

1.2.3 Security Metric: Reliability 5

1.2.4 VLSI Metrics: Area, Power, Delay 5

1.2.5 Testchips 6

1.3 Outline of Thesis 6

2 PUF Implementations and Design Guidelines **9**

2.1 Generation of a Random Bit 9

2.2 PUF Design vs. a Typical Digital Design 11

2.3 PUF Implementations 11

2.3.1 Delay PUFs 12

2.3.2 Bi-Stable PUFs 15

2.4 Strong-PUFs vs. weak-PUFs 19

2.5 Summary of Chapter 20

3 Comparison of PUF Implementations 21

3.1 Arbiter PUFs 22

3.1.1 Test structure on chip 22

3.1.2 Loss of entropy as a function of arbiter bias 22

3.2 Ring Oscillator PUF 22

3.2.1 Test structure on chip 22

3.2.2 RO PUF measurements from a variety of ROs 25

3.3 Bi-stable PUFs: SRAM and sense amplifiers 30

3.3.1 Test structure on chip 30

3.4 Comparison of security metrics 30

3.4.1 Uniqueness 30

3.4.2 Randomness 32

3.4.3 Reliability 38

3.5 Comparison of VLSI Metrics 44

3.5.1 Area 44

3.5.2 Delay 44

3.5.3 Energy 46

3.6 Summary of Comparison 47

3.7 Reliability, a Hard Problem 47

3.7.1 Conventional Error Correction 48

3.7.2 Error Reduction Techniques 51

3.8 Summary of Chapter 51

4 Extrinsic Techniques for Reliability Enhancement 53

4.1 Multiple Evaluation (ME) 54

4.1.1 Introduction 54

4.1.2 Delay-based PUFs 54

4.1.3 Bi-stable PUFs 55

4.2 Activation Control (AC) 60

4.2.1 Introduction 60

4.2.2 SRAM PUFs: VDD Power-up Ramp Control 60
4.3 Summary of Chapter 65

5 Intrinsic Reliability Enhancement by Post-Silicon Selection (PSS) 66
5.1 PSS in Sense Amplifier PUFs 69
5.1.1 Sense Amplifier Offset Voltage 69
5.1.2 SA Offset Voltage as a Reliability Estimator 70
5.1.3 Efficient Reliability Characterization 70
5.1.4 Reliability Enhancement (Results from Generation I Testchip) 72
5.2 Autonomous Key Generator and Secure Strong-PUF 76
5.2.1 Design 76
5.2.2 Experimental Results (Generation II testchip) 80
5.2.3 Reliability of Generated Key 80
5.2.4 Error Modeling 85
5.2.5 Uniqueness 85
5.2.6 Randomness of the Selected Bits 85
5.2.7 Speed 88
5.2.8 Area 90
5.3 PSS in SRAM and delay-based PUFs 91
5.4 Discussion of Results 91
5.4.1 Strong PUF 92
5.5 Summary of Chapter 92

6 Intrinsic Reliability Enhancement by Directed Accelerated Aging (DAA) 94
6.1 Aging Phenomena used for DAA 95
6.1.1 Negative Bias Temperature Instability (NBTI) 95
6.1.2 Hot Carrier Injection (HCI) 96
6.2 DAA using NBTI in SRAM PUFs (NBTI-DAA) 96
6.2.1 Example Procedure 98
6.2.2 Reliability Enhancement: Experimental Results 99

6.2.3 NBTI-DAA as an Attack 99
6.2.4 Implementation Options 101
6.3 DAA using HCI in SA PUFs (HCI-DAA) 102
6.3.1 HCI-Enabled Sense Amplifier (HCI-SA) PUF Structure 103
6.3.2 Use case 107
6.3.3 Shift in Offset by HCI Stress: Experimental Results 107
6.3.4 Reliability Enhancement: Experimental Results 111
6.3.5 Permanence of Reliability Reinforcement 113
6.3.6 Uniqueness and Randomness of HCI-SAs 114
6.4 Comparing DAA-NBTI and DAA-HCI 116
6.5 DAA in Other PUF Types 117
6.6 Discussion of Results 117
6.7 Summary of Chapter 118

7 Conclusion and Future Work **120**

7.1 Summary 120
7.2 Conclusions 121
7.3 Future Research Directions 121

A Description of Testchips **123**

A.1 Generation I Testchip 123
A.2 Generation II Testchip 124

B Sense Amplifier Offset Measurement Methodology **131**

B.1 Sense Amplifier Array 131
B.2 Offset Measurement 131

C Test Infrastructure **133**

C.1 Design 133
C.2 Testing 133

Bibliography **134**

List of Figures

1.1 A typical PUF system. 2
1.2 PUF Uniqueness 3
1.3 PUF Randomness 5
1.4 Organization of thesis. 7

2.1 Simplified illustration of the distribution of the generated bit by a PUF core by amplification of some electrical property 10
2.2 Two delay paths with same nominal delays but different random components of delays. 12
2.3 Simplified arbiter PUF 13
2.4 A N-stage arbiter PUF that generates a 1-bit response 14
2.5 Simplified ring oscillator PUF 14
2.6 A typical RO PUF implementation 15
2.7 A bi-stable cross-couple 16
2.8 A typical 6T SRAM cell. The bi-stable cross-couple is shown in blue. 16
2.9 Sense amplifier PUFs 17

3.1 Layout of arbiter PUF test structures 23
3.2 Modified arbiter PUF test structures 24
3.3 Impact of arbiter bias on entropy 24
3.4 Modified ring oscillator PUF test structure 25
3.5 Layout of ring oscillator PUF test structures 26
3.6 Measurements from a variety of RO PUFs (continued...) 27

3.6 (...continued) Measurements from a variety of RO PUFs 28

3.7 Bi-stable PUFs 31

3.8 Layout of ring oscillator PUF test structures 31

3.9 Layout of ring oscillator PUF test structures 32

3.10 Measured uniqueness across chips for various PUF implementations 33

3.11 Measure of randomness: bias across chips for different PUF types 38

3.12 Randomness: Bias across layout orientations (mean across chips) 39

3.13 Randomness: Bias across layout orientations (all chips) 40

3.14 Measure of randomness: Histogram of #1s across rows/columns for SRAM and SA PUFs 41

3.15 Reliability of various PUF implementations 43

3.16 Area comparison 45

3.17 Delay comparison 46

3.18 Energy comparison 47

3.19 PUF comparison summary 48

4.1 Greyscale bitmap showing power-up values of SRAM and SA array with multiple evaluations 56

4.2 Errors in initial response bitmap after N evaluations compared to a "golden bitmap" generated after 999 evaluations 57

4.3 Errors with multiple evaluations in-field 58

4.4 Simulated waveform of SRAM internal nodes during VDD power-up 61

4.5 Reliability of SRAM with ramp rate when golden and in-field evaluation is done at same ramp rate 62

4.6 Reliability of SRAM with ramp rate when golden bitmap is generated at a fixed ramp rate 63

5.1 A simplified illustration of post-silicon selection (PSS) 67

5.2 Distribution of measured offset of sense amplifiers 69

5.3 Measured V_{OFFSET} of a sample of sense amplifiers across voltage and temperature variations. 71
5.4 Reliability pre-characterization of SA PUFs 73
5.5 Reliability of SA PUF using post-silicon selection (Generation I Testchip Results) 74
5.6 Percentage Utilization-Reliability tradeoff with PSS in SA PUFs 75
5.7 Key generator and strong-PUF design 77
5.8 Layout snapshot of key generator and AES for strong PUF realization. 78
5.9 Flowchart of key generation and strong-PUF realization. 79
5.10 Reliable bit aggregator 80
5.11 Reliability bitmap generation in the key generator 82
5.12 % of SA-PUF bits selected after reliability pre-characterization 83
5.13 Reliability of bits from key generator 84
5.14 Modeling errors in bits from the key generator 86
5.15 Uniqueness of bits from key generator across dies 87
5.16 Percentage of 1's in the selected bits for various ΔV_{IN} for 3 different chips. . . . 88

6.1 NBTI phenomenon in PMOS devices 95
6.2 Stressing a NMOS device by hot carrier injection (HCI) 97
6.3 How NBTI affects a 6T SRAM cell 98
6.4 Biasing SRAM cells toward 1 via NBTI-DAA 100
6.5 Error reduction in SRAM PUF using NBTI-DAA 101
6.6 Schematic of HCI-enabled sense amplifier (HCI-SA) PUF 103
6.7 Layout of HCI-SA PUF 104
6.8 Steps of operation of HCI-SA. 106
6.9 Offset voltage of all HCI-SAs in ascending order 108
6.10 Scatter plot of shift in offset of HCI-SAs after stress 109
6.11 Changing distribution of sense amplifier offset with HCI-stress 110
6.12 Reliability of HCI-SA PUFs after HCI-stress 112
6.13 Permanence of reliability reinforcement (offset shift due to HCI-stress) 115
6.14 Uniqueness of HCI-SAs 116

A.1 Dieshot of Generation I Testchip 125
A.2 Top-level layout snapshot of PUF structures on Generation I testchip 126
A.3 Printed circuit board (PCB) to test Generation I Testchip 128
A.4 Dieshot of Generation II Testchip 129
A.5 Top-level layout snapshot of PUF structures on Generation II testchip 129
A.6 Printed circuit board (PCB) to test Generation II Testchip 130

List of Tables

3.1 NIST randomness test results for delay based PUFs. 36
3.2 NIST randomness test results for bi-stable based PUFs. 36

5.1 Area of the key generator 90

A.1 Generation I Testchip 124
A.2 Description of 256 ROs on a die 127
A.3 Generation II Testchip 127

Chapter 1

Introduction

1.1 What is a PUF?

A Physical Unclonable Function or a PUF is a die-specific random function or a *silicon biometric* that is unique for every instance of the die. PUFs derive their randomness from the uncontrolled random variations in the IC manufacturing process (usually undesirable) to create practically unclonable functions even if the original design files are compromised.

PUFs are increasingly used as building blocks in many secure systems for applications such as device identification/authentication [12, 13, 16, 17, 19, 23, 25, 27, 30, 31, 36, 65, 66] and secret key generation [1, 2, 4, 19, 36, 38, 63, 65, 67]. PUFs provide an attractive alternate to *storing* of random secret bits in volatile or non-volatile memory (which are vulnerable to attacks [45]) by instead *generating* these random bits every time the PUFs are evaluated.

1.1.1 Challenge and Response

An input to a PUF is typically called the *challenge* and the generated output is typically called the *response*. An applied challenge and its response are generally called a *challenge-response pair* or a CRP. These terms are from the field of security, where challenge-response authentication is a process that verifies an identity by requiring correct authentication information in response to a challenge.

1.1.2 Silicon PUFs and Non-Silicon PUFs

In this work, we focus only on the silicon based PUF designs which are designed and fabricated just like any other CMOS circuit, allowing easy integration with existing silicon based security system designs. Note that other non-silicon based PUF implementations have been proposed in literature, e.g., optical PUFs [52], but difficulties in integration with standard CMOS design and manufacturing flow limit their practicality.

1.1.3 The PUF Core

At the core of a silicon PUF is a circuit (the PUF core) that generates random bits. The PUF core circuit leverages the typically small and uncontrolled random manufacturing process variations to generate die-specific, unique, random set of bits. A PUF derives all its entropy from these underlying PUF core bits and hence designing good PUF core bits is imperative in achieving a PUF of high quality. Figure 1.1 shows how a PUF core circuit may be embedded in a PUF system for the generation of the output (response) when provided with an input (challenge). As shown in Figure 1.1, peripheral circuits may implement input or output scramblers to condition the challenge and response for added security. Error correction may be implemented to correct errors in the raw response from the PUF core. The response of a PUF should satisfy the following security properties:

- Unique: die-specific, like a silicon biometric.
- Random: difficult/impossible to model the response.
- Reliable: consistent across environmental variations and aging.

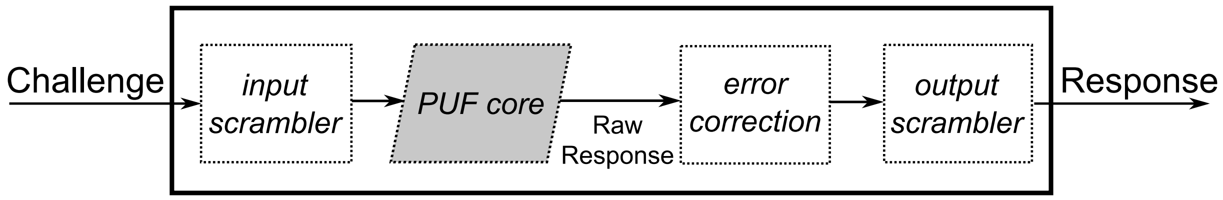


Figure 1.1: A typical PUF system generates a response when provided with a challenge. The PUF core is the source of entropy for the PUF system. The challenge and response may be conditioned using scramblers. Error correction may be implemented to increase the reliability of the generated response.

In this thesis, as in most of the literature, the term "PUF" is often used to describe both the "PUF core" that generates a single response bit as well as a complete system as shown in Figure 1.1.

1.2 Physical Unclonable Functions: Metrics

In this section we discuss the figures of merit of a PUF implementation. We first discuss the security metrics of uniqueness, randomness, and reliability followed by the conventional VLSI metrics of area, power, and performance.

1.2.1 Security Metric: Uniqueness

Uniqueness (or inter-die randomness) is a measure of how uncorrelated the response bits are across dies, and ideally the response bits should differ with a probability of 0.5. It is a measure of how different a PUF instantiation on a die is from another such instantiation built using the same design files and technology (Figure 1.2). Since each bit on a die should differ from that on another with a probability of 0.5, the Hamming distance of a multi-bit response (say N=32) across dies should follow a binomial distribution with parameters N=32 (response width) and p=0.5. High uniqueness can be achieved by minimizing any systematic bias in the design that can lead to a predictable or modelable bit response across chips. Various studies in literature have demonstrated that a reasonably high degree of uniqueness across dies is achievable for most PUF implementations [9, 13, 16, 17, 25, 26, 31, 39, 65, 67].

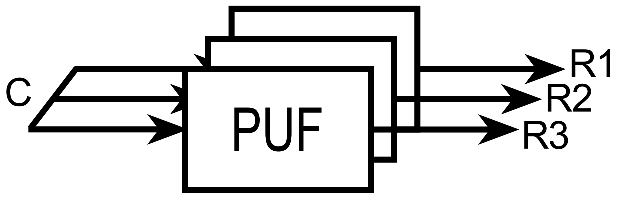


Figure 1.2: PUF Uniqueness: The response (R) should be different/unique for every instantiation of a PUF even when created from the same design files, and when provided with the same challenge(C).

1.2.2 Security Metric: Randomness

Randomness (or intra-die randomness) is a measure of the unpredictability of the response. This implies (i) unpredictability of a response for a new challenge despite the prior knowledge of a large number of challenge-response pairs (CRPs) (Figure 1.3) as well as (ii) unpredictability of every bit in the response even with a knowledge of all other response bits. Providing guarantees about randomness is often non-trivial in practice. Various test suites exist that aim to find a “pattern” (and hence non-randomness) in a sequence of bits under test [21, 29, 49, 58]. Note that although a failure to find patterns in any set of tests can suggest randomness from a statistical point of view, no set of finite empirical tests can guarantee absolute randomness. For this reason, simply the application of a randomness test suite is not sufficient to verify the randomness of the responses of a PUF. The design factors that may result in a bias in the PUF response must therefore be carefully analyzed. A bias in the response of nominally equivalent PUFs is a result of a systematic bias in the variations that are leveraged for the generation of the response bits. An analysis of the potential systematic biases in the design may provide a better insight in designing a PUF with high randomness. In this work, we analyze the randomness of PUF response in the following two ways:

- Measuring any bias in the response (i.e., percentage of 1’s and percentage of 0’s) in the entire response data set and in various slices of the data set. The slices are created to find layout or structural dependent biases.
- Running the standard National Institute for Science and Technology (NIST) statistical tests [58] on the responses.

High randomness can be achieved by using circuits with regular layout and minimum sized devices. Regularity minimizes the impact of intra-die systematic variation [24] and small sizing maximizes the impact of random variations [54] (e.g. random doping fluctuations and line edge roughness). Various studies in literature have demonstrated that reasonably high degree of randomness across dies is achievable for most PUF implementations [9, 13, 25, 26, 39, 65, 67].

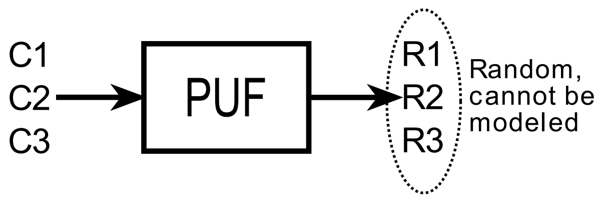


Figure 1.3: PUF Randomness: when a PUF is provided with different challenges (*C*), the responses (*R*) that are generated should be random.

1.2.3 Security Metric: Reliability

Reliability is a measure of repeatability or consistency with which a PUF generates its response across environmental variations, ambient noise, and aging. Although most of the response bits of a PUF demonstrate high reliability, some (typically ∼1-20%, depending on the PUF type and evaluation conditions) do not have a strong bias towards resolving to either of the two states. High reliability is desirable in all applications but is absolutely critical in applications such as encryption key generation. The conventional method to improve reliability is to use powerful error correction codes (ECC) to correct the raw response from the PUF core [73]. Unfortunately, these ECC blocks generally have high VLSI overheads, which scale up quickly as the error correction capability increases. Moreover, some of the error correction techniques require careful implementation to avoid introduction of any security vulnerability through leakage of information about the response bits [73]. Researchers have also proposed the use of several orthogonal low-overhead error *reduction* techniques that will minimize the use of high-overhead error *correction* techniques, resulting in a system with overall lower overheads and/or higher security [10, 69].

1.2.4 VLSI Metrics: Area, Power, Delay

The conventional VLSI metrics of a VLSI design are area, power, and performance. Although these metrics are often secondary to the security metrics for a PUF design, many PUF applications may impose strict area, power, and/or performance requirements. Area usually has a direct impact on the cost of die and a smaller area may enable the use of PUFs even in low-cost applications. Several implementations, e.g. RFID, are severely power-limited requiring PUFs to operate under a power budget. Higher performance may be required for high-performance cryptographic applications. This study compares the area, power, and delay of the several PUF implementations.

1.2.5 Testchips

All the results in this work are supported by measured silicon results. For this we fabricated two testchips, both in 65nm bulk CMOS. A description of the testchips and the included test-structures is given in Appendix A.

1.3 Outline of Thesis

In this thesis, we discuss various aspects of a PUF design, with a strong focus on circuit and implementation details. The outline of this work is shown in Figure 1.4.

In Chapter 2 and Chapter 3, we discuss various PUF implementations and provide a comparison on security and VLSI metrics using hardware results. Based on the comparison, we conclude that while randomness and uniqueness is achievable for most PUF implementations, achieving high reliability has remained a challenge. We briefly discuss the two methods of achieving high reliability. The conventional method is to use error correction codes (ECC) and significant amount of work has been done in this field in the last decade [11, 14, 20, 37, 40, 44, 53, 73, 74]. We investigate the other method of reliability enhancement by error *reduction*. We categorize the error reduction techniques as *extrinsic* and *intrinsic*.

In Chapter 4, we discuss two extrinsic techniques - multiple evaluation (ME) and activation control (AC). These techniques show limited improvements in reliability. Measured results show that they are able to reduce errors by 70-80%.

In Chapter 5, we describe our first intrinsic technique – post-silicon selection (PSS). The technique, when applied to sense amplifier PUFs demonstrate $\sim$100% reduction in errors. We designed a self-contained BIST-controlled key generator system which autonomously generates highly reliable bits that can be used as a key. The measured failure rate of a 128-bit key using this technique was $< 10^{-6}$. We also describe the framework of a secure and reliable strong-PUF

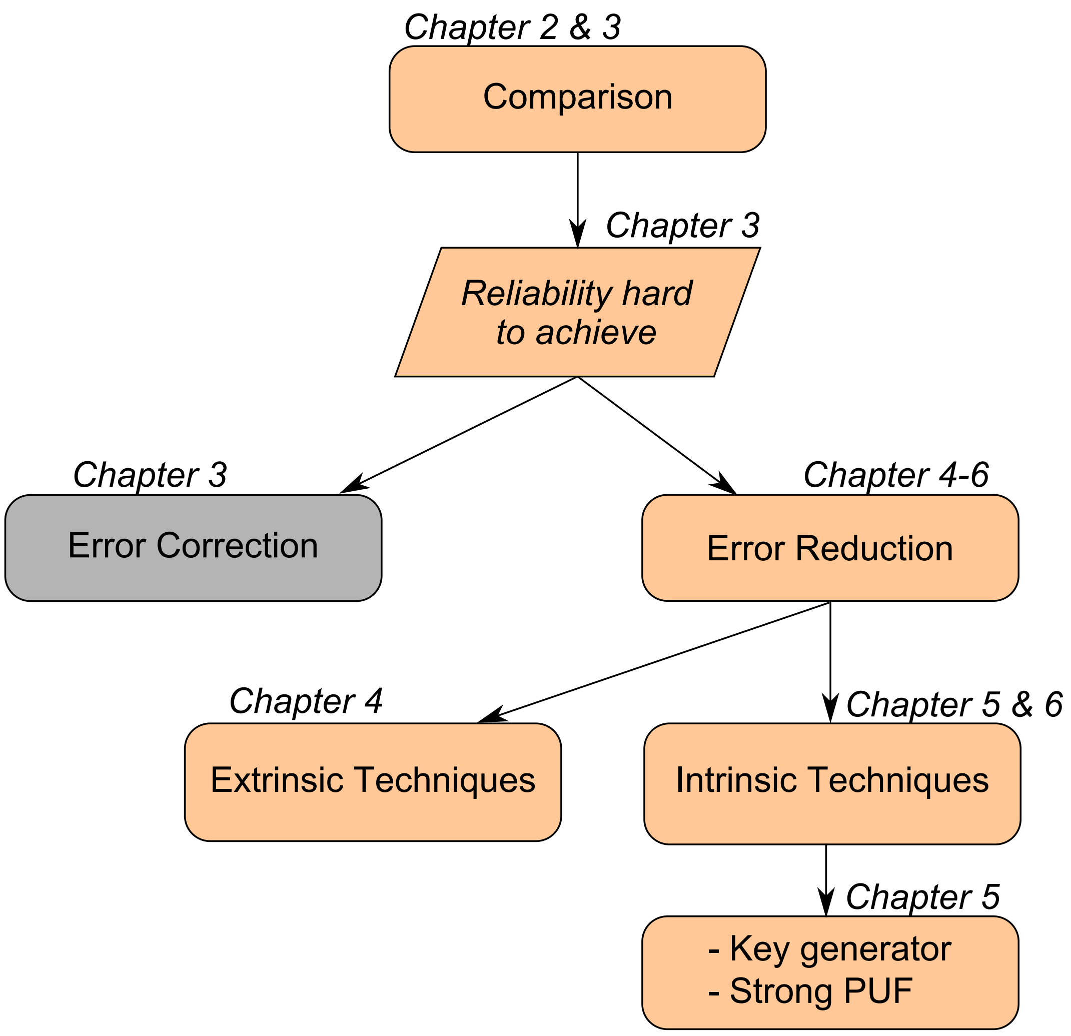


Figure 1.4: Organization of thesis.

that uses the reliable bits from the key generator in conjunction with an AES primitive.

In Chapter 6, we describe our second intrinsic technique – directed accelerated aging (DAA). We show how IC aging mechanisms can be used in a directed and accelerated manner to increase the PUF reliability. Using the HCI-based DAA on a modified sense amplifier PUF, we were able to show $\sim 100\%$ error reduction. Measured results show that this technique is able to generate bits with an equivalent 128-bit key error rate $< 10^{-6}$.

In Chapter 7, we summarize the work and draw conclusions.

Chapter 2

PUF Implementations and Design Guidelines

2.1 Generation of a Random Bit

Over the last decade, several PUF core implementations have been proposed in literature. A typical PUF core implementation generates its random bits by amplifying the difference in some electrical characteristic of two identically designed circuits. Most PUF implementations can be categorized as either *delay-based* or *bi-stable element based*, depending on this electrical characteristic. Delay-based PUFs (or simply delay-PUFs) compare the nominally identical delay paths to determine a response bit [16, 65]. Bi-stable element-based PUFs (or simply bi-stable PUFs) use the activation state of a nominally balanced bi-stable element to determine a response bit [8, 26]

For illustration, let us assume $\mathcal{P}$ is an electrical property of two identically designed circuits in a PUF core (measured as $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$) such that the difference $\mathcal{D} = \mathcal{P}^{(1)} - \mathcal{P}^{(2)}$ is amplified for the generation of a random bit in a PUF core. Since the property $\mathcal{P}$ can have both randomly ($\mathcal{P}_R$) and systematically ($\mathcal{P}_S$) varying components, their difference $\mathcal{D}$ will have a random ($\mathcal{D}_R = \mathcal{P}_R^{(1)} - \mathcal{P}_R^{(2)}$) and a systematic component ($\mathcal{D}_S = \mathcal{P}_S^{(1)} - \mathcal{P}_S^{(2)}$).

Let us assume that the randomly varying component ($\mathcal{P}_R$) has a normal distribution $\mathcal{N}(\mu_{\mathcal{P}_R}, \sigma_{\mathcal{P}_R})$. Hence, $\mathcal{D}_R$ will have a normal distribution $\mathcal{N}(0, \sqrt{2}\sigma_{\mathcal{P}_R})$. Let us also assume (pessimistically

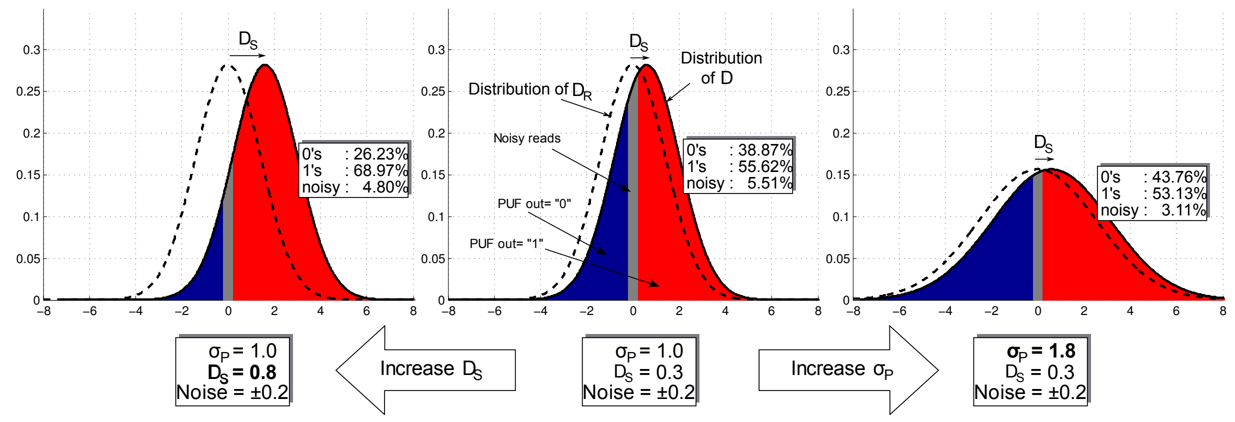


Figure 2.1: (center) Probability density function of $\mathcal{D}_R$ and $\mathcal{D}$, where $\mathcal{D}$ is the difference in an electrical property $\mathcal{P}$ when measured from two identically designed half-circuits in a PUF core. This difference $\mathcal{D}$ is amplified for the generation of a random bit. A systematic component ($\mathcal{D}_S$) will result in a bias in the generated random bit (and hence a lower than ideal entropy). The grey region represents noisy PUF outputs. (center) A typical scenario with $\sigma_\mathcal{P}$ = 1 a.u. ($\sigma_\mathcal{D} = \sqrt{2}$ a.u.), small $\mathcal{D}_S$ (= 0.3 a.u.), and noisy region represented by $\pm$0.2 a.u. (right) A wider distribution of $\mathcal{P}$ results in a reduced number of noisy outputs, even for the same $\mathcal{D}_S$ and noisy region. (left) A larger systematically varying component results in a increased bias in the PUF output.

from a security perspective) that the systematically varying component is completely understood by an adversary and $\mathcal{D}_S$ can be accurately modeled. For simplicity, assume that $\mathcal{D}_S$ is a constant. Then $\mathcal{D}$ will have a normal distribution $\mathcal{N}(\mathcal{D}_S, \sqrt{2}\sigma_{\mathcal{P}_R})$. Figure 2.1 shows the distribution of $\mathcal{D}$, the quantity amplified for the generation of a random bit. The amplification of $\mathcal{D}$ for the generation of the PUF response bit is assumed to be unbiased. However, due to the resolution limit of the amplification process and due to any differential impact of ambient noise on $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$, circuits with $\mathcal{D} \approx 0$ will have non-deterministic (noisy) outputs. These noisy outputs are represented by the grey region around the region $\mathcal{D} = 0$. Therefore, bits that are generated by amplifying a larger value of $\mathcal{D}$ are expected to be more reliable (consistent). Statistically, this is shown in Figure 2.1 (right) where a wider distribution of $\mathcal{P}$ (and hence $\mathcal{D}$) results in a lower percentage of noisy bits. Also Figure 2.1 (left) shows that a small value of $\mathcal{D}_S$) would result in a smaller bias, and hence a random bit with higher entropy.

This simple analysis suggests the following.

- For high randomness, the systematic component of $\mathcal{D}$ needs to be minimized. This minimizes any bias in the response and hence maximizes entropy. This can be achieved by high symmetricity in the design of the PUF core.

- For high reliability, the spread of $\mathcal{D}$ needs to be maximized to ensure that a larger number of random bit generating PUF cores are unaffected by environmental variations and ambient noise and regenerate bits consistently. This can be achieved by using minimum sized devices to maximize the impact of variations.

Please note the above analysis, if done accurately on any PUF circuit will be much more complex. For example, $\mathcal{P}$ may not follow a normal distribution or it may not be possible to model $\mathcal{D}_S$ accurately as assumed in this analysis. This will result in quantitative differences, however, the qualitative design guidelines will still hold true.

2.2 PUF Design vs. a Typical Digital Design

Silicon PUFs, just like typical digital circuits, generate an output (response) every time they are provided with an input (challenge). Moreover, they have the same design flow and process flow like any typical digital circuit. Yet, there are some fundamental differences in the way they are designed. A digital circuit is typically designed with enough margin such that its output does not depend on process variations and is entirely a deterministic function of its input. On the other hand, a PUF is designed such that its output is entirely a function of the process variations and is not a predictable function of its input at the time of design.

As described in the previous section, PUFs generate their random outputs by amplifying some electrical property that is measured from two nominally identical circuits. It is important to note that this difference in the electrical property is not inherent in the PUF design, but is only due to process variations.

2.3 PUF Implementations

Most of the PUF implementations proposed in literature can be classified as either delay PUFs or bi-stable PUFs, depending on the phenomenon of response generation. In this section, we discuss the design and properties of two representative designs of the delay PUFs and bi-stable

PUFs.

2.3.1 Delay PUFs

A delay-based PUF compares the delay of two nominally identical paths. The basic idea is as shown in Figure 2.2. The two delay paths Path1 and Path2 have the same nominal delay $T_{nominal}$. However, their delays, T1 and T2, are different because of different random delay components $T_{random1}$ and $T_{random2}$. A response bit in a delay PUF is generated by comparing (directly or indirectly) T1 and T2, which effectively is comparing $T_{random1}$ and $T_{random2}$. If $T_{random1}$ and $T_{random2}$ are randomly and identically distributed, the response bit generated by comparing T1 and T2 will be a random bit. Arbiter and ring-oscillator PUFs are the most common implementations of delay based PUFs.

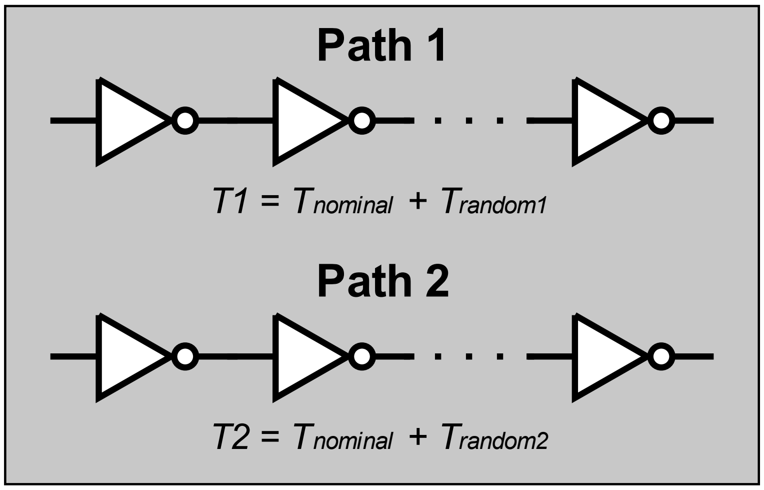


Figure 2.2: Two delay paths with same nominal delays but different random components of delays.

Arbiter PUFs. Arbiter PUFs are delay-based PUFs and generate their response bit by comparing the delay of two delay paths, nominally identical, but with slightly different delays due to manufacturing process variations. The basic idea is as shown in Figure 2.3. The paths in an arbiter PUF connect to a decision-making arbiter block, essentially a phase comparator, that generates a 0 or 1 depending on which path is faster.

However, in practice, to allow a large challenge-response space, the inverters in the delay

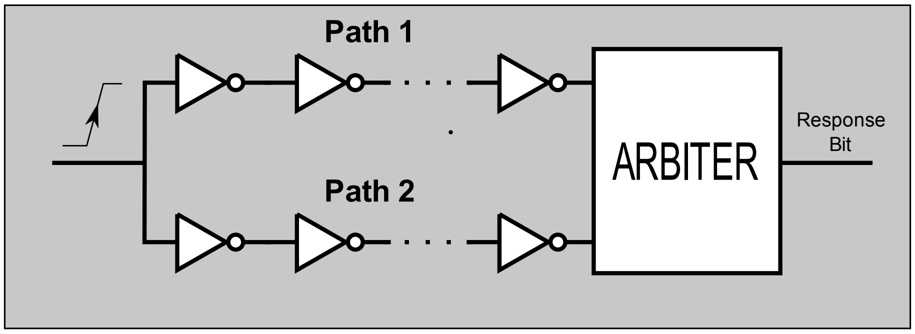


Figure 2.3: Simplified arbiter PUF

paths are replaced by configurable multiplexers as shown in Figure 2.4 [16]. The arbiter can be implemented using digital latches or D-flops. Ideally, an arbiter PUF should be able to generate 2^N unbiased and random response bits for each of the possible 2^N combinations of challenges. However, arbiter PUFs have two known issues.

1. **Vulnerability to modeling attacks:** Because of the additive nature of the delays of the various stages, an arbiter PUF is vulnerable to modeling attacks. Several modifications to the arbiter (e.g., feed-forward arbiter PUF) have been proposed to improve resistance to modeling attacks [34, 36, 48]. Unfortunately, researchers have demonstrated that simple machine learning algorithms can be used to build accurate models even in the modified designs [36, 38, 47, 57]. These modeling attacks are typically able to learn from around 5000-10000 CRPs to predict the response to a new challenge with an accuracy $>95\%$.

2. **Bias in responses:** Since a single arbiter PUF chain (as shown in Figure 2.4) is challenged multiple times to generate responses, a bias in the arbiter will result in a bias in all the generated bits from that chain.

Ring Oscillator (RO) PUFs. Ring oscillator (RO) PUFs compare the frequency of two nominally identical ring oscillators for the generation of a response bit. The basic idea is as shown in Figure 2.5. The frequency of the selected ROs is counted using a counter followed by a comparison of the counter values for the generation of a response bit.

A typical implementation is shown in Figure 2.6 [65] and consists of multiple identical ROs, followed by two wide muxes to select the two ROs for comparison. Conceptually, an RO PUF works in a way that is very similar to an arbiter PUF. Where an arbiter PUF has a phase com-

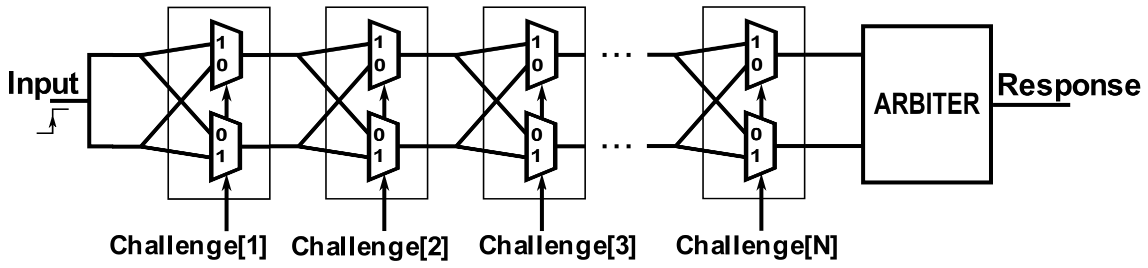


Figure 2.4: A Typical arbiter PUF with two nominally identical delay paths followed by an arbiter to generate a response bit depending on which path is faster.

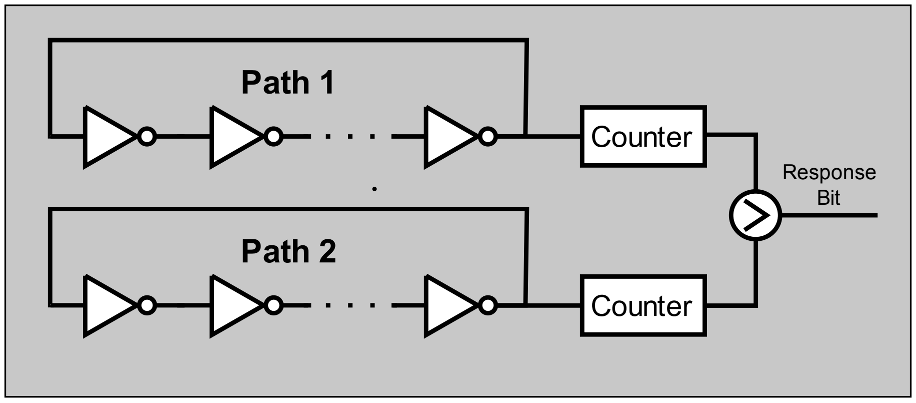


Figure 2.5: Simplified ring oscillator PUF

parator (i.e., the arbiter) to amplify the difference of a pair of delay paths, the RO PUF amplifies the difference of the frequency of the selected ROs by allowing them to oscillate for a time that is long enough for the frequency measuring counters to generate at least one bit of difference.

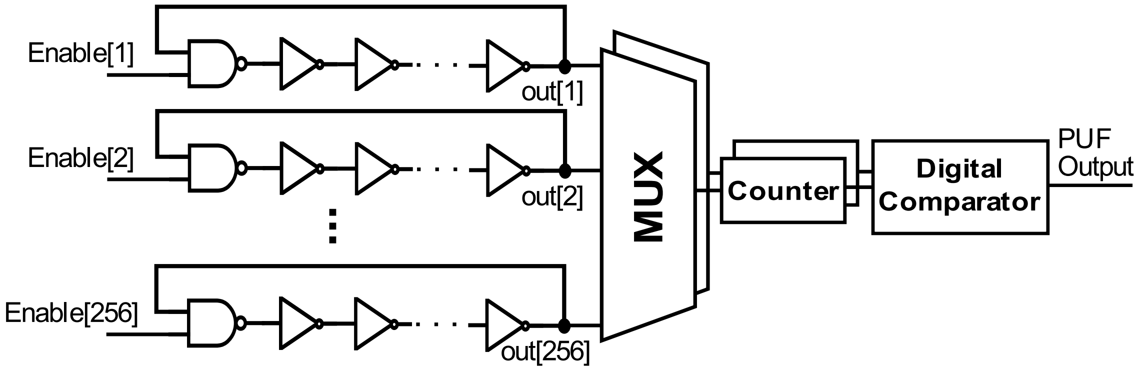


Figure 2.6: A typical implementation of ring oscillator PUF with multiple rings followed by two wide muxes to select two rings for comparison. The frequency of the selected rings is counted in the two counters followed by a comparison of the counter values for the generation of a response bit.

2.3.2 Bi-Stable PUFs

Bi-stable PUFs, as implemented by a cross-couple as shown in Figure 2.7 has two stable states by construction: A=1 or A=0. To generate a random bit, the cross-couple is reset by forcing A and B to be equi-potential and then allowing them to resolve. If the matched devices in the cross-couple are sized and laid out symmetrically (i.e. MN1 and MN2 are symmetrically designed and MP1 and MP2 are symmetrically designed), then the final state of the cross-couple is equally likely to be A=1 or A=0 and depends on the differences in the devices strengths that originate from process variations. If the process variations are pre-dominantly random in nature, the final state of the cross-couple is random and unpredictable.

SRAM PUFs. SRAM PUFs have a bi-stable, symmetrically designed cross-couple element as shown in blue Figure 2.8. The 6T SRAM cell also has two pass-gates (also called access devices) that are used to read from and write to the cross-couple. To use as PUF, SRAMs are typically completely discharged (i.e., supply voltage VDD = voltage at A = voltage at B = 0V) and then the supply voltage VDD is slowly ramped up.

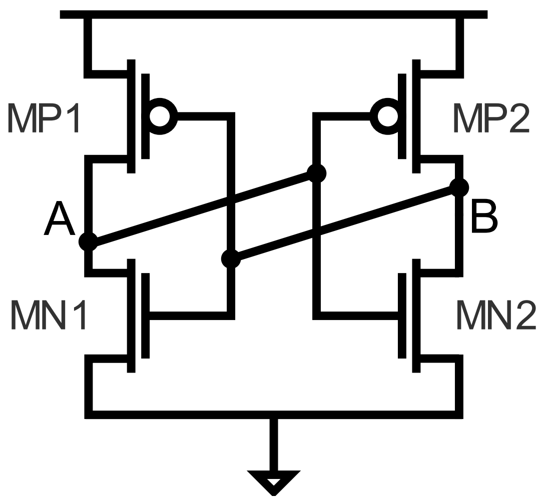


Figure 2.7: A bi-stable cross-couple with two stable states: (A,B)=(0,1) or (1,0).

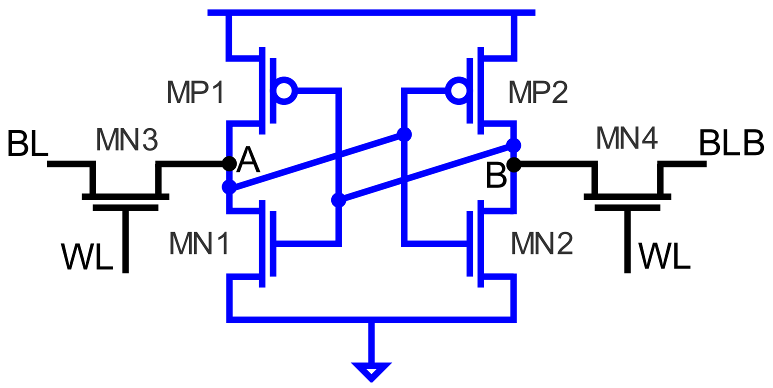
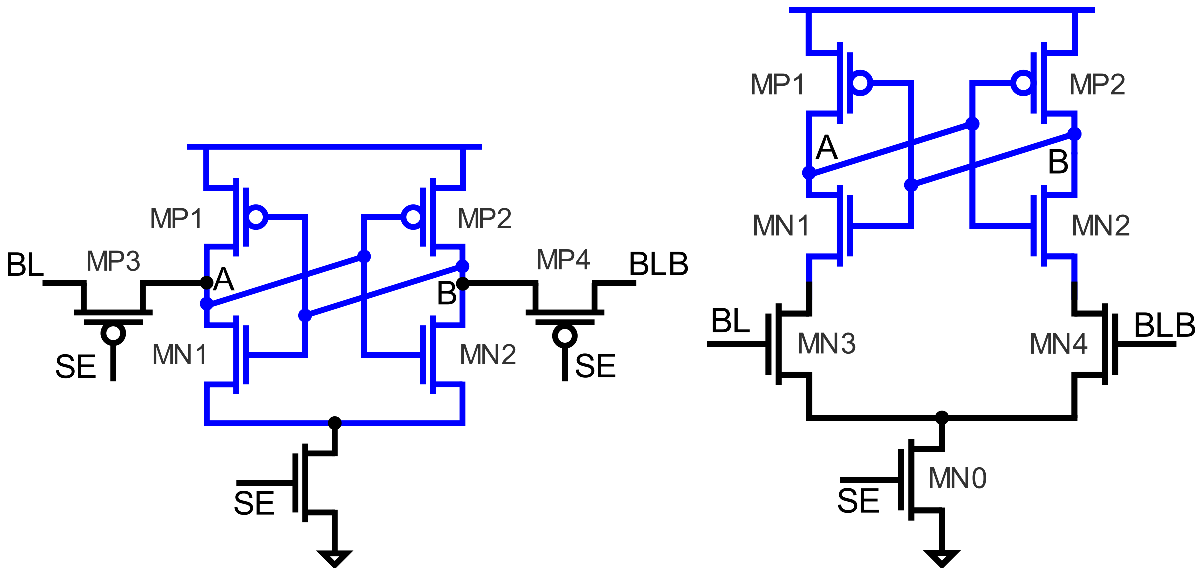


Figure 2.8: A typical 6T SRAM cell. The bi-stable cross-couple is shown in blue.

Sense Amplifier (SA) PUFs. Sense amplifiers (SAs) are clocked circuits that amplify small differential voltages into full swing digital values, and are commonly used in memory read paths and as voltage comparators. Figure 2.9 shows the schematic of two popular topologies of sense amplifiers - the latch style and the StrongARM. Under ideal conditions, an ideal SA would correctly amplify even the smallest of input differential voltages. In practice, however, variations in the devices of an SA may result in an offset (or bias), a measure of the natural tendency of the SA to resolve to a particular polarity. To ensure correct operation, the SA inputs need to have a difference larger a certain voltage, referred to as the offset voltage or V_{OFFSET}.



(a) Latch style sense amplifier **(b)** StrongARM sense amplifier

Figure 2.9: (a) A latch style sense amplifier. (b) A StrongARM sense amplifier. The bi-stable cross-couple is shown in blue.

To use as a PUF core, SA inputs (BL and BLB in Figure 2.9) are shorted together (i.e., set to the same voltage, zero differential input) and the SA is fired (SE goes high). A SA will then resolve to a value determined by its individual V_{OFFSET} [8]. We shall see later in Section 5 that the V_{OFFSET} of a SA has a strong correlation with its reliability and a SA with higher $|V_{OFFSET}|$ is more likely to be reliable. The V_{OFFSET} of the latch style SA (Figure 2.9(a)) is a strong function of the difference in threshold voltage (V_{TH}) of devices MN1 and MN2 and the V_{OFFSET} of the StrongARM SA (Figure 2.9(b)) is a strong function of the differences in the

V_{TH} of MN1 and MN2 as well as the differences in the V_{TH} of MN3 and MN4. The $|V_{OFFSET}|$ (and hence the reliability) of a SA can be increased by increasing the difference in the V_{TH} of these pairs of devices. This can be achieved by using near-minimum sized devices to maximize the variations in the V_{TH} of these devices [54]. Moreover, to minimize systematic component of offset, i.e., maximize the randomness of the response, these pairs of devices in the SA should be designed as symmetrically as possible.

Offset of a SA results from a combination of systematic and random variations. Systematic variations can be due to manufacturing gradients and layout asymmetries [5], and can be minimized by symmetric layout of matched devices. Random variations are a result of random uncertainties in the fabrication process such as random dopant fluctuation (fluctuations in the number and location of dopants in the transistor channel) [28] and gate line-edge roughness [51]. The effects can be mitigated by using larger devices [54].

Other bi-stable PUFs. Many other PUF implementations exist in literature that can be categorized as bi-stable PUFs. All the designs have a cross-coupled structure which is reset and then evaluated to generate a response bit. Some of the main proposed implementations are as follows.

- **Latch PUF:** The settling state of two cross-coupled NAND (or NOR) gates which constitute a simple SR latch has been proposed as a PUF core [64]. The cross-couple is a symmetric structure with nominally equivalent gates and equal capacitances and hence it can generate unbiased responses with 50% 1's and 50% 0's.
- **Flip-Flop PUF:** The power-up state of a Flip-Flop has been proposed as a PUF core [43, 68]. The cross-coupled structure is inherently asymmetric – the cross-coupled gates are typically a simple inverter and a tri-stated inverter with unequal strengths, and the capacitance of the cross-coupled nodes (equivalent to A and B in Figure 2.7) is also not matched. Hence, the power-up state of a Flip-Flip PUF can have a bias to either "1" or "0". This was confirmed from large-scale silicon measurements where the uniqueness across bits generated from a Flip-Flop PUF was only 36% (as opposed to an ideal value of 50%) [68].
- **Butterfly PUF:** The settling state of two cross-coupled latches has been proposed as a PUF core in FPGAs [33]. For FPGAs, ensuring symmetric capacitance of the cross-couple

nodes is not easy and hence Butterfly PUFs can have generated bits with a strong bias to either “1” or “0”.

- **Buskeeper PUF:** The settling state of a buskeeper i.e., an uncontrolled latch implemented as cross-coupled inverters with weak drive strength, has been proposed as a PUF core [61]. Buskeeper PUFs can result in a biased response due to inherent mismatch in the capacitance of the cross-coupled nodes.

2.4 Strong-PUFs vs. weak-PUFs

The distinction between a strong-PUF and a weak PUF was introduced in [20] and was later formally defined in [56]. Basically, the distinction is based on the security properties of challenge-response pairs (CRPs). Simply put, a PUF is called a strong-PUF if it has a large (often exponential to some system parameter) number of CRPs. The weak PUFs, on the other hand, can only generate a few CRPs. From a security perspective, this means that for a strong-PUF, an adversary cannot predict the response of a random challenge with high probability, even with the prior knowledge of a large number of CRPs. Most PUF implementations, including RO PUF, SRAM PUF, and SA PUF, only generate a small number of random bits and are therefore classified as weak PUFs. Although, the arbiter PUF can be challenged by an exponentially large number of inputs for the generation of response bits, it has been shown that simple machine learning algorithms can be used to build accurate models for the response behavior with the knowledge of a small number of CRPs ($\sim$5000-10000) [36, 38, 47, 57].

Properties of a truly secure and reliable strong-PUF have been discussed in [41, 56] and designing a silicon strong-PUF is generally considered an open problem. In Section 5.2, we propose and describe the framework of a strong-PUF that uses our proposed reliable key generator in conjunction with an AES block, a standard cryptographic primitive. Our strong-PUF leverages the randomness properties of the AES to generate one-way, non-invertible responses. The reliability of the key generator ensures that the strong-PUF is reliable.

2.5 Summary of Chapter

In this chapter we have discussed how a typical PUF implementation generates its random bits by amplifying some electrical property from two nominally identical circuits in the PUF core. We provide a simplified illustration to show how characteristics of this electrical property impacts the security metrics of a PUF. We then discuss the design and properties of the two main categories of PUF implementations, the delay PUFs and bi-stable PUFs. There are several studies in literature that evaluate the security and/or VLSI metrics of a single PUF core implementation. A comparison of various PUF implementations that is based on the results from these studies is both difficult and inaccurate because of differences in metrology and process technologies used.

In the next chapter, we provide an apples-to-apples comparison of the security and VLSI metrics (as discussed in Section 1.2) of several PUF implementations that we fabricated on the same platform.

Chapter 3

Comparison of PUF Implementations

As discussed in Section 2, PUFs derive their security properties from the bits generated by the underlying PUF core. There are several studies in literature that evaluate the security and/or VLSI metrics of a single PUF core implementation. However, using the results from different studies for a comparative analysis is not possible for various reasons. First, these studies do not use a standard test methodology or a common definition of the metrics for comparison. Second, some of the published results are from simulation-based analysis while some are from silicon-based measurements. Even for the publications that provide silicon measurements, some are from FPGA based designs and some are from ASIC designs. Third, the results are from different CMOS process technologies and/or using different manufacturers/technology models making it even harder to compare the results across studies.

Hence, in order to provide an apples-to-apples comparison, we designed a testchip that contained several PUF implementations on the same platform. The testchip is built in 65nm bulk CMOS technology and is described in Appendix A.1 (Generation I testchip). In this chapter we describe the test structures that we built on our Generation I testchip followed by a comparative analysis of the security and VLSI metrics from the measured results.

3.1 Arbiter PUFs

3.1.1 Test structure on chip

As described earlier in Section 2.3.1, arbiter PUFs are delay-based PUFs and generate their response bit by comparing the delay of two nominally identical delay paths. On the testchip we built 12 instances of arbiter delay chains, each consisting of two configurable delay paths, in a total area of 42kμm^2 (Figure 3.1). The delay paths are configured using a 64-bit challenge. The test-structures are a slight modification of the original proposed arbiter PUF structure in that the final arbiter block is removed and the chains are looped back on themselves, which allows them to oscillate (Figure 3.2). Measurement of the oscillation frequency enables precise path delay measurement and phase detection with arbitrary resolution. The modification also enables the study of loss in entropy as a function of a bias in the phase detector.

3.1.2 Loss of entropy as a function of arbiter bias

A large bias in the phase detector will result in a significant loss in entropy. This is of special concern when the same chain, and hence the same biased phase comparator is re-used for response generation for multiple challenges (as is proposed in [36, 38]). Figure 3.3 shows the loss in entropy as a function of a bias in the phase comparator. The delay difference in the chains (rings in the modified design) are measured on a testchip at nominal conditions (1.2V and 27°C). Arbiter PUFs have been shown to be vulnerable to modeling attacks [57] and a lowered entropy of the response bits potentially makes them even more vulnerable to such attacks.

3.2 Ring Oscillator PUF

3.2.1 Test structure on chip

As described in Section 2.3.1, RO PUFs compare the frequency of two nominally identical ring oscillators for the generation of a response bit. The test structures on the testchip were modified from the typical implementation by replacing the two wide muxes, the two counters, and the

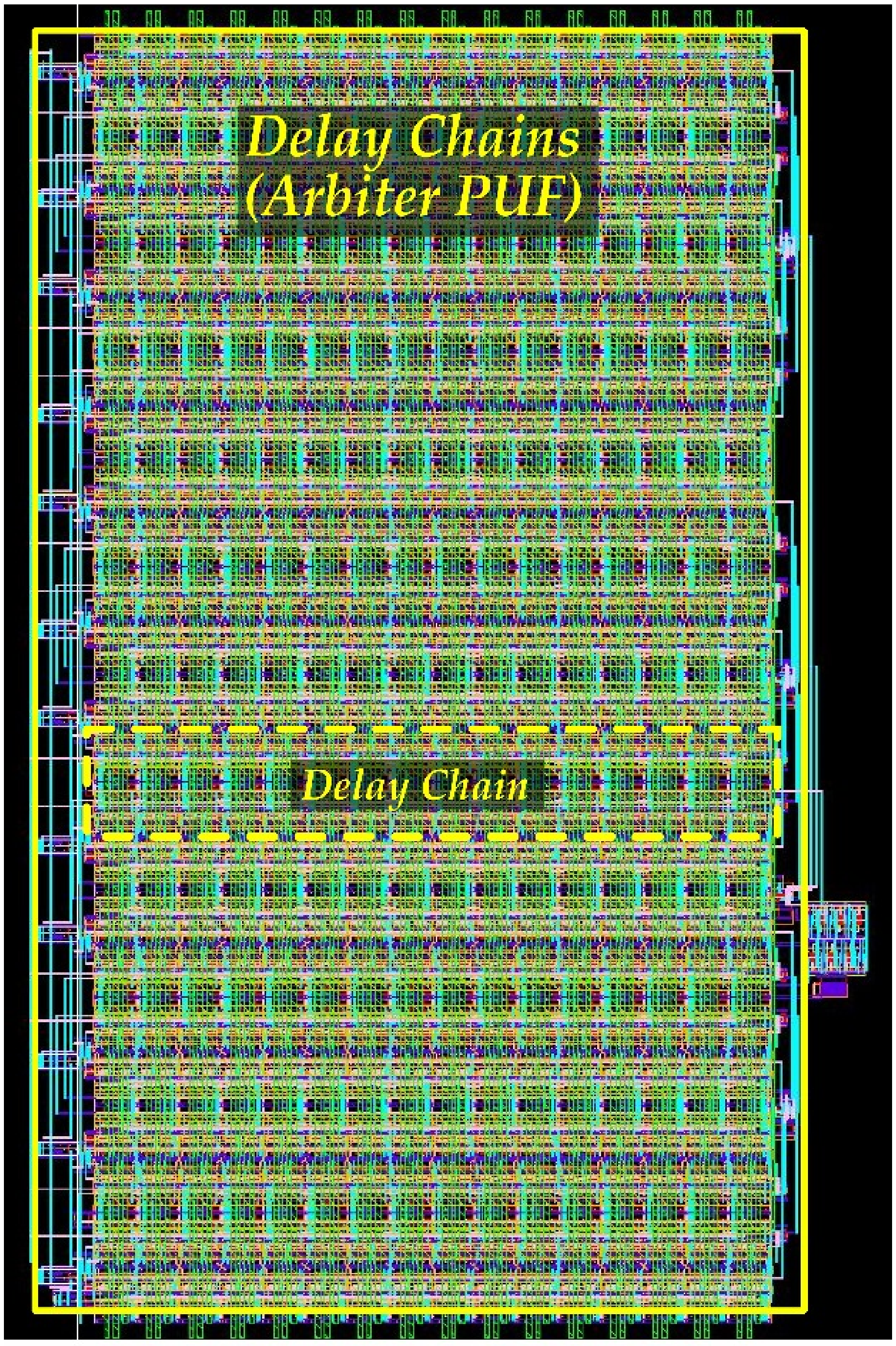


Figure 3.1: Layout of arbiter PUF test structures

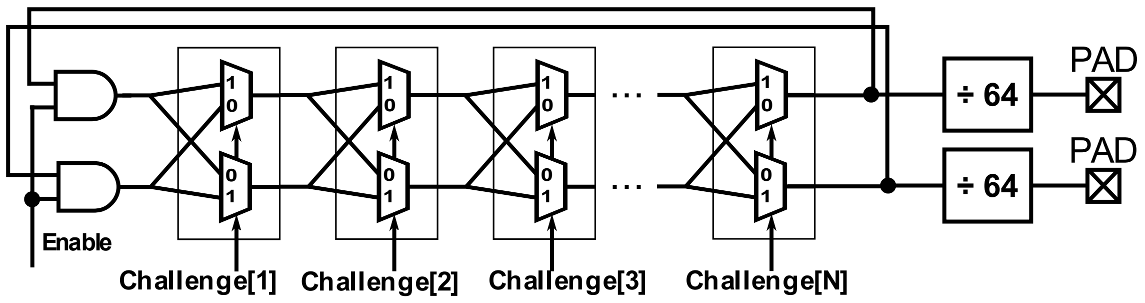


Figure 3.2: Modified arbiter PUF test structure on the testchip with the final arbiter block removed and the delay chains looped back on themselves, allowing them to oscillate for precise delay measurements.

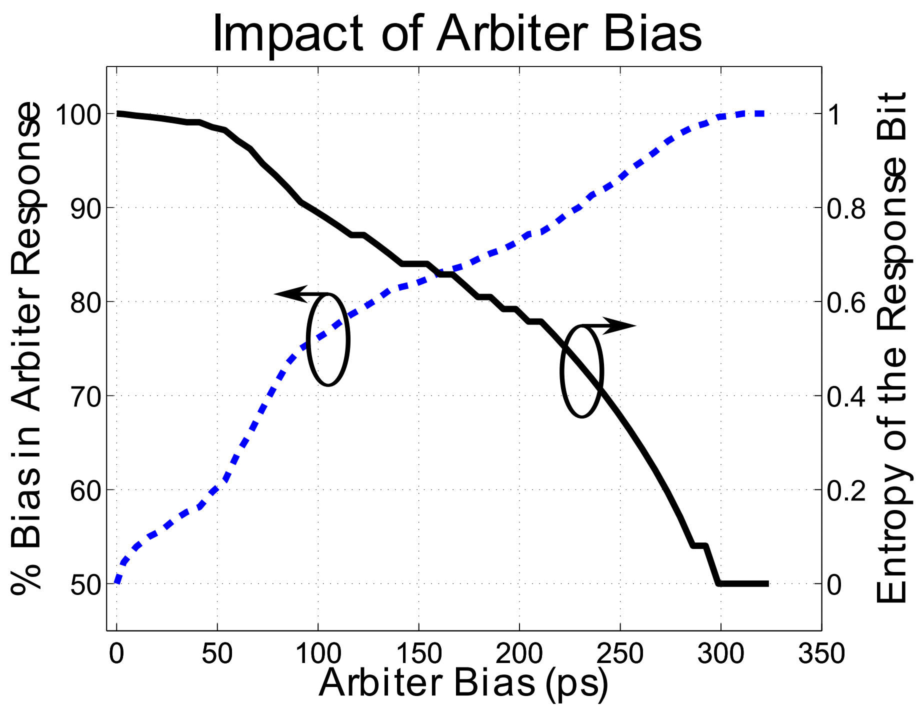


Figure 3.3: Arbiter PUF loss of entropy as a function of a fixed bias in the phase comparator. Ideal % bias and ideal entropy of the response bit is 50% and 1 respectively.

comparator by a single wide mux followed by a frequency down converter (Figure 3.4). This signal after the frequency down conversion enables accurate off-chip frequency measurements. The testchip contains 256 ROs built using a variety of ring sizes (13, 17, and 31 inverter chains) and device types (all combinations of 'Low Power' and 'General Purpose' process devices with high, low, and standard V$_{TH}$ doped devices) in an area of 30kμm^2 (Figure 3.5). Measurements from different design options are presented next.

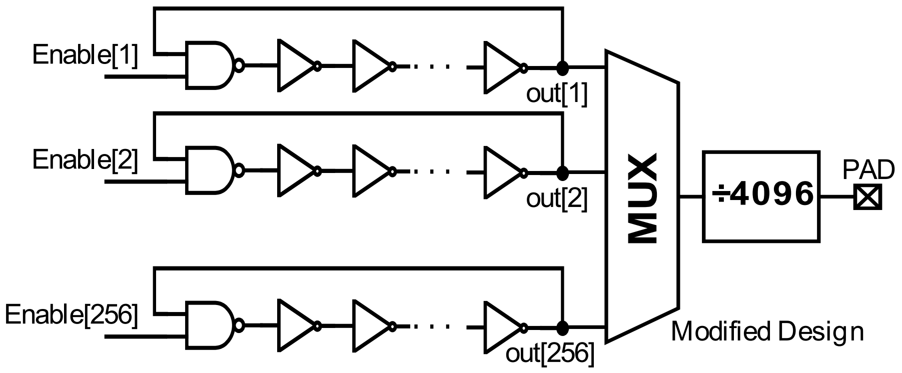


Figure 3.4: Modified ring oscillator PUF test structure on the testchip. The test structures are modified by replacing the two wide muxes, the two counters, and the comparator by a single wide mux followed by a frequency down converter to measure the oscillations accurately off-chip.

3.2.2 RO PUF measurements from a variety of ROs

A description of all the 256 rings on a die is provided in Table A.2. Relevant results from the rings is presented here. Figure 3.6(a) shows the frequency of 32 ROs built using GPsvt devices (i.e., GP process and standard-V$_{TH}$ doping) across temperature. Figure 3.6(b) and 3.6(c) show the mean frequency of ROs built using various types of devices across VDDs (at nominal temperature 27°C) and across temperature (at nominal VDD 1.2V for LP, 1.0V for GP) respectively. Figure 3.6(d) shows mean cycle time (i.e., 1/frequency) of ROs of different lengths. Figure 3.6(g) and 3.6(h) show the standard deviation of frequency across VDDs (at nominal temperature) and across temperature (at nominal VDD) respectively. Figure 3.6(e) and Figure 3.6(f) show the standard deviation of the cycle time and frequency, respectively, of ROs of different lengths.

Figure 3.6(a), 3.6(b), and 3.6(c) show that, as expected, the frequency of ROs increase for

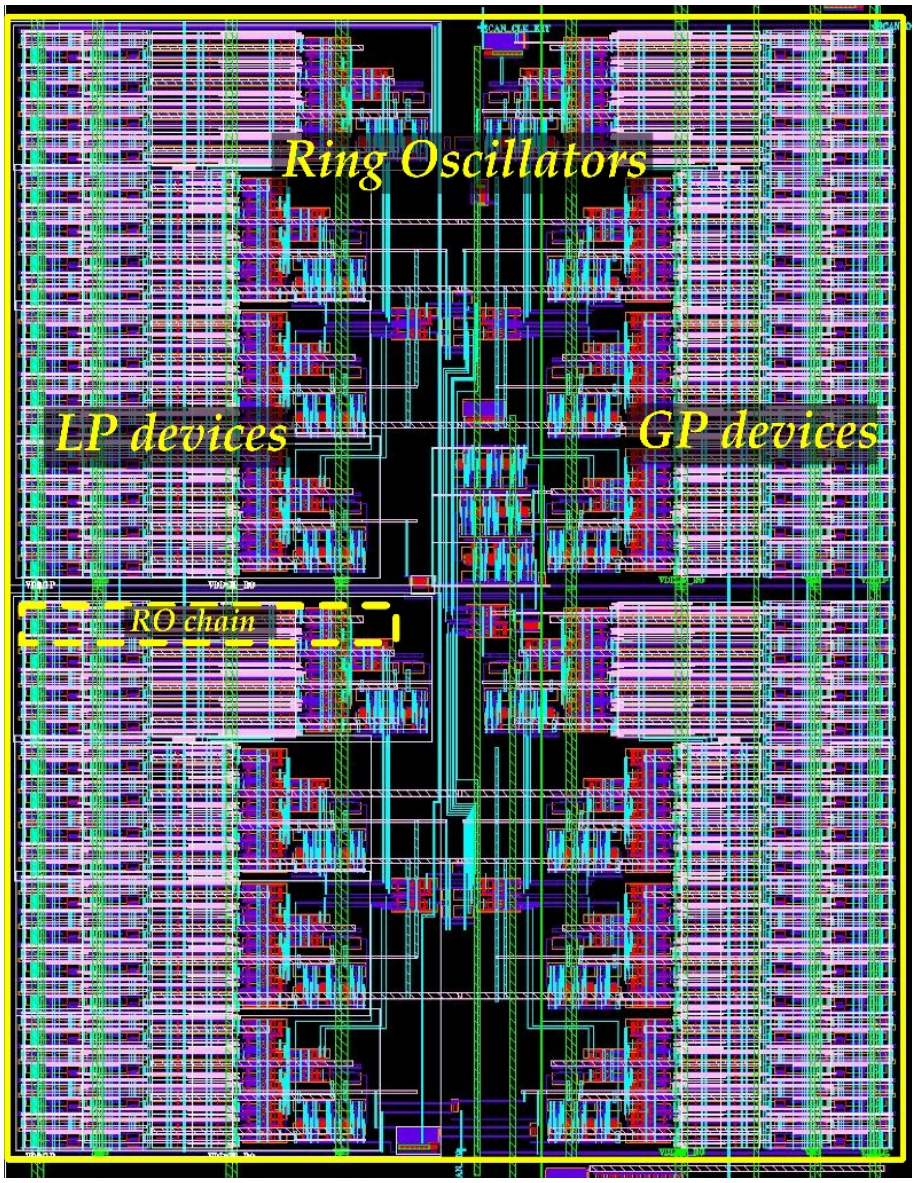
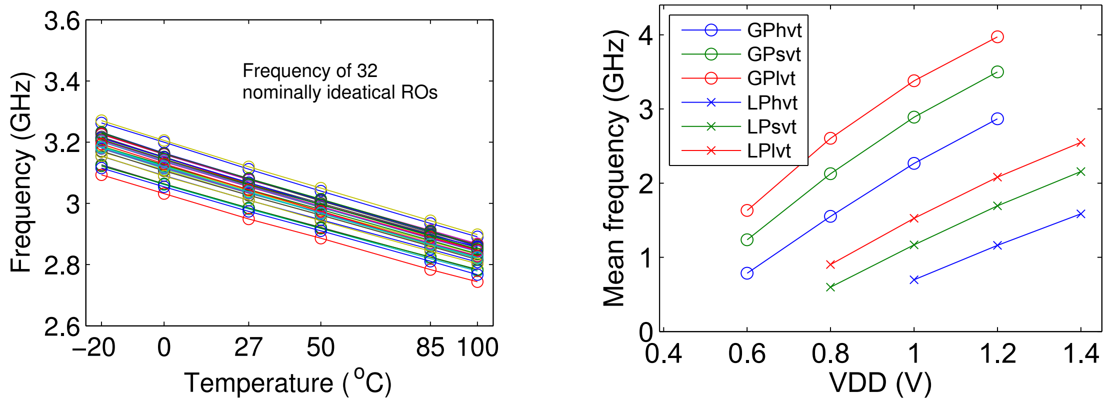


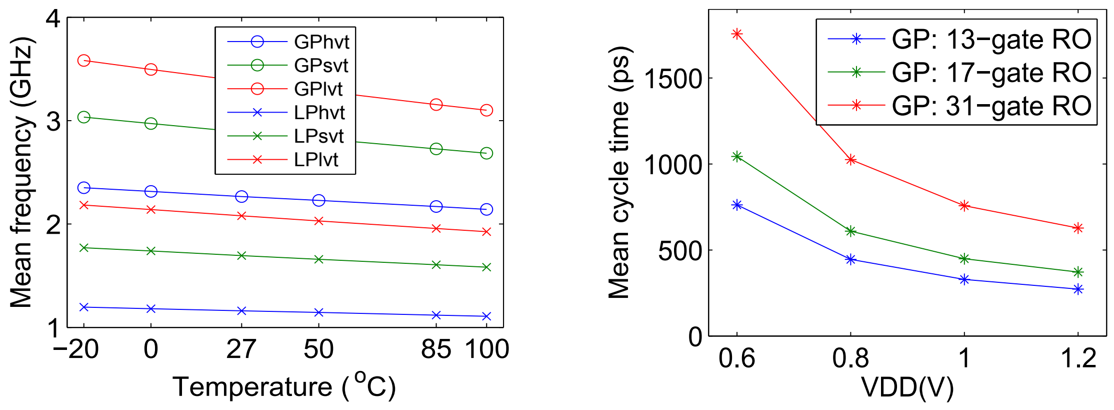
Figure 3.5: Layout of ring oscillator PUF test structures

higher VDD and at lower temperatures. Moreover, as expected ROs built using GP process are faster than those built using LP process. Further, within a process option, the low-V$_{TH}$ devices (lvt) have a higher frequency compared to standard-V$_{TH}$ devices (svt) followed by high-V$_{TH}$ devices (hvt).



(a) Freq of 32 ROs (GPsvt) across T (VDD=1V)

(b) Mean freq across VDD (27°C)

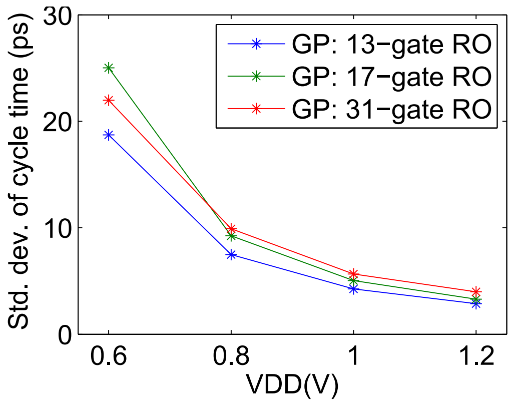


(c) Mean freq across T (VDD=1V)

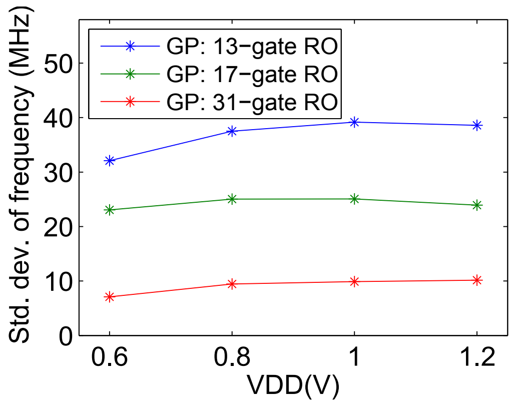
(d) Mean cycle time (1/freq) across VDD (27°C)

Figure 3.6: Measured results from various ROs. (a),(b),(c) show measurements from rings designed using a variety of device types. (continued on next page...)

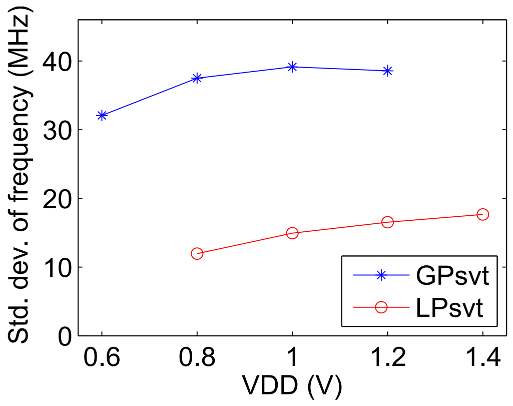
Figure 3.6(d) shows ROs of longer lengths take more time to oscillate and have a higher cycle time. The delay of a RO can be approximated to be the sum of the delay of all the individual gates in the rings. Under this approximation, we would expect the mean cycle times of the 17-gate RO and the 31-gate RO to be 17/13 times (=1.31x) and 31/13 times (=2.38x) respectively of that of the cycle time of the 13-gate RO. For ROs built in LP, the ratios were found to be ∼1.33 and ∼2.31 and for ROs built in GP, the ratios were found to be ∼1.36 and ∼2.30. From this we can



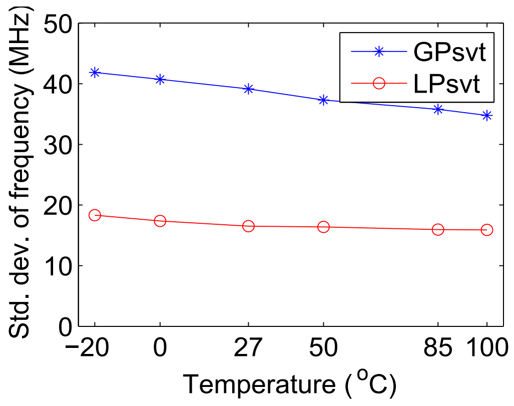
(e) Std. dev. of cycle time across VDD (27°C)



(f) Std. dev. of freq across VDD (27°C)



(g) Std. dev. of freq across VDD (27°C)



(h) Std. dev. of freq across T (VDD=1V)

Figure 3.6: (...continued from previous page) Measured results from various ROs. (d),(e),(f) show measurements from rings of different lengths. (g) and (h) show the dependency of standard deviation of frequency (and hence the evaluation time of RO PUFs) on VDD and temperature.

conclude that RO delays can be modeled as the additive delay of the individual gates in the ring. Additive delay approximation would suggest that the standard deviation of cycle times of the 17-gate RO and the 31-gate RO would be $\sqrt{17/13}$ times (=1.14x) and $\sqrt{31/13}$ times (=1.54x) respectively, under the assumption that each gate delay is independent and identically distributed with a normal spread. The measured ratios, however, deviated from these numbers. For example, the ratios for rings built in GP and delays measured at nominal voltage and temperature were found to be 1.18 and 1.33. One reason for this could be that we have only 16 rings of size 17 and 31 per die and empirically measured standard deviation requires much larger number of samples to converge as compared to the mean.

Figure 3.6(f), 3.6(g), and 3.6(h) show measurements of the standard deviation of the measured frequency for various cases. The evaluation time of an RO PUF is the time a pair of rings is expected to take to generate at least a 1-bit difference in their frequency count and hence the evaluation time is a function of the frequency difference of the selected rings. The larger the spread of the frequency of a sample of rings, the better the chances of having a large difference in the frequency of the two arbitrary selected rings. In other words, the time to wait for rings with larger spread is less and hence RO PUFs that use rings with large frequency distribution have faster evaluation times. The RO PUF evaluation time is inversely proportional to the standard deviation of the frequency of the rings.

From Figure 3.6(f), we can conclude that RO PUFs that use smaller rings have faster evaluation times. From Figure 3.6(g) and Figure 3.6(h), we can conclude that RO PUFs built using GP devices have faster evaluation time than ones built using LP devices. Figure 3.6(b) show that by increasing the supply voltage from 0.8V to 1.2V for GPsvt rings and from 1.0V to 1.4V for LPsvt rings, the mean frequency of the GPsvt rings and LPsvt rings increase by 63% and 81% respectively. However, for the same increase in voltage, the standard deviation of frequency of GPsvt and LPsvt rings increases (and hence the RO PUF evaluation time decreases) by only 3% and 18% respectively (Figure 3.6(g)). Since the dynamic power is approximately proportional to the square of VDD, increasing VDD will result in a significant increase in dynamic power while the RO PUF evaluation time improves only slightly. Hence, this suggests that RO PUFs should be evaluated at lower voltage for an optimal power-delay product. Figure 3.6(h) also shows that the RO PUF evaluation time not have a strong dependency on the temperature of operation.

3.3 Bi-stable PUFs: SRAM and sense amplifiers

3.3.1 Test structure on chip

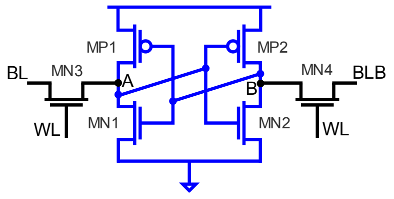
As described in Section 2.3.2, SRAM and sense amplifier (SA) PUFs have bi-stable elements that evaluate to one of the two stable states. The SRAM PUF is typically activated by powering-up the supply. The SA PUF is activated by firing the sense enable. As seen in Figure 3.7, the SRAM and SA implementations have essentially the same core circuit (cross coupled inverter pair). For the SRAM, the source of the PMOS pair is directly controlled whereas for the SA implementations, the source of the NMOS pair is controlled via a gate-controlled NMOS device. The testchip contains two 64x64 SRAM arrays in a total area of 21.0kμm^2 (Figure 3.8) and two 64x64 SA arrays in a total area of 26.9kμm^2 (Figure 3.9), one using latch-style SAs and one using StrongARM SAs. The SA arrays are arranged and accessed like the SRAM array. When used as a PUF, the inputs are shorted and a row of SAs is triggered by sense enable (implemented like a wordline in a SRAM).

3.4 Comparison of security metrics

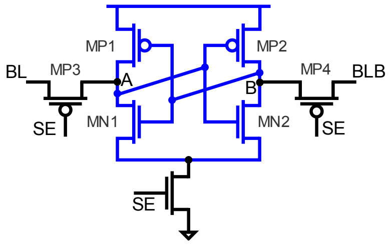
In this section we discuss how the PUF core bits generated from different PUF implementations evaluate on the security metrics of uniqueness, randomness, and reliability.

3.4.1 Uniqueness

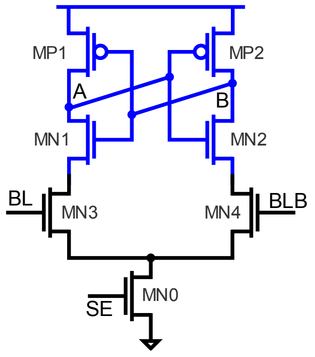
Uniqueness (or inter-die randomness) of a PUF is a measure of how uncorrelated the response bits are across dies and is typically evaluated by measuring the distribution of Hamming distance of multi-bit responses across dies. Ideally, the distribution should follow a binomial distribution with parameters N = response width and p = 0.5. Figure 3.10 shows the histogram of Hamming distance of PUF responses from two chips for all PUFs. The response for the SRAM and



(a) 6T SRAM



(b) Latch style sense amplifier



(c) StrongARM sense amplifier

Figure 3.7: 6T SRAM, latch-style, and StrongARM SA schematics. In the shown implementations, SRAM PUFs are typically activated by raising the power supply after the internal nodes are completely discharged. Sense amplifier PUFs are activated by a rising sense enable signal (SE) after the internal nodes are reset. The core of all cells is a cross-couple inverter bi-stable (shown in blue) with very similar mode of operation. Where the SRAM is VDD referenced, the sense amplifier imlementations shown are GND referenced.

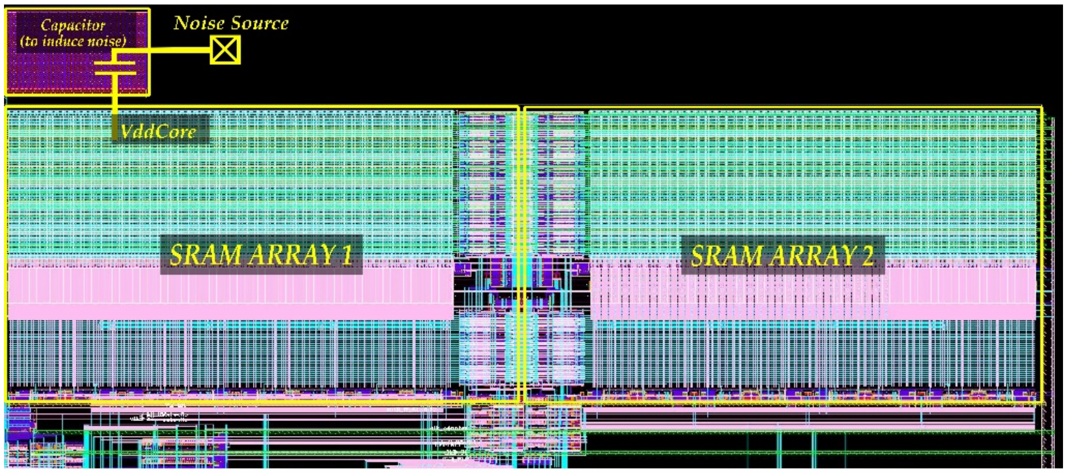


Figure 3.8: Layout of ring oscillator PUF test structures

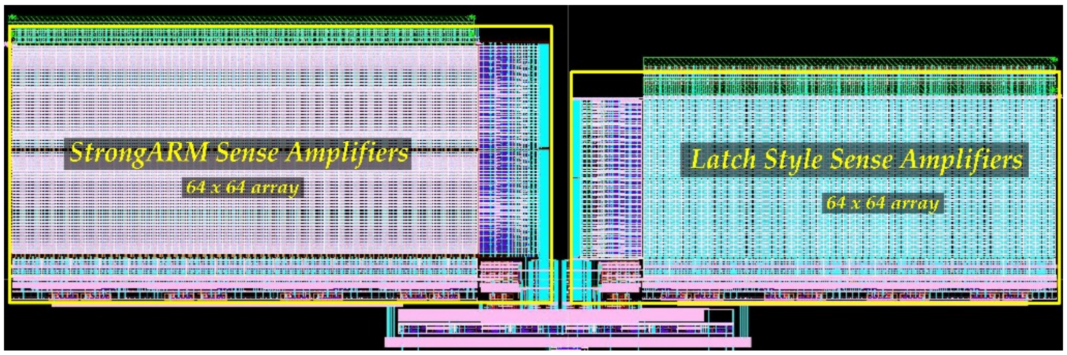


Figure 3.9: Layout of ring oscillator PUF test structures

SA PUFs were created using a 16-bit word (4 responses per row). A single arbiter chain was randomly challenged 16 times to create a 16-bit response. The small number of ROs of the same design type allowed a maximum of 4-bit response width for the RO PUF. The responses for all the PUFs were found to be close to the behavior of perfectly unique PUFs. Measurements from all other pairs of chips showed similar results.

3.4.2 Randomness

Randomness (or intra-die randomness) is a measure of the unpredictability of the response. This implies (i) unpredictability of a response for a new challenge despite the prior knowledge of a large number of challenge-response pairs (CRPs) as well as (ii) unpredictability of every bit in the response even with a knowledge of all other response bits. Many test suites like the NIST test suite [58], Knuth's empirical tests [29], the Diehard test suite [49], and Gustafson's tests [21] exist to statistically analyze the randomness of a binary sequence. These randomness tests can be used to identify patterns (and hence non-randomness) in the response bits. Note that although a failure to find patterns in any set of tests can suggest randomness from a statistical point of view, no set of finite empirical tests can guarantee absolute randomness. The NIST test suite document [58] states that the statistical tests may be a "useful first step" but "cannot serve as a substitute for cryptanalysis." For this reason, simply the application of a randomness test suite is not sufficient to verify the randomness of the responses of a PUF. The design factors that may result in a bias in the PUF response must therefore be carefully analyzed.

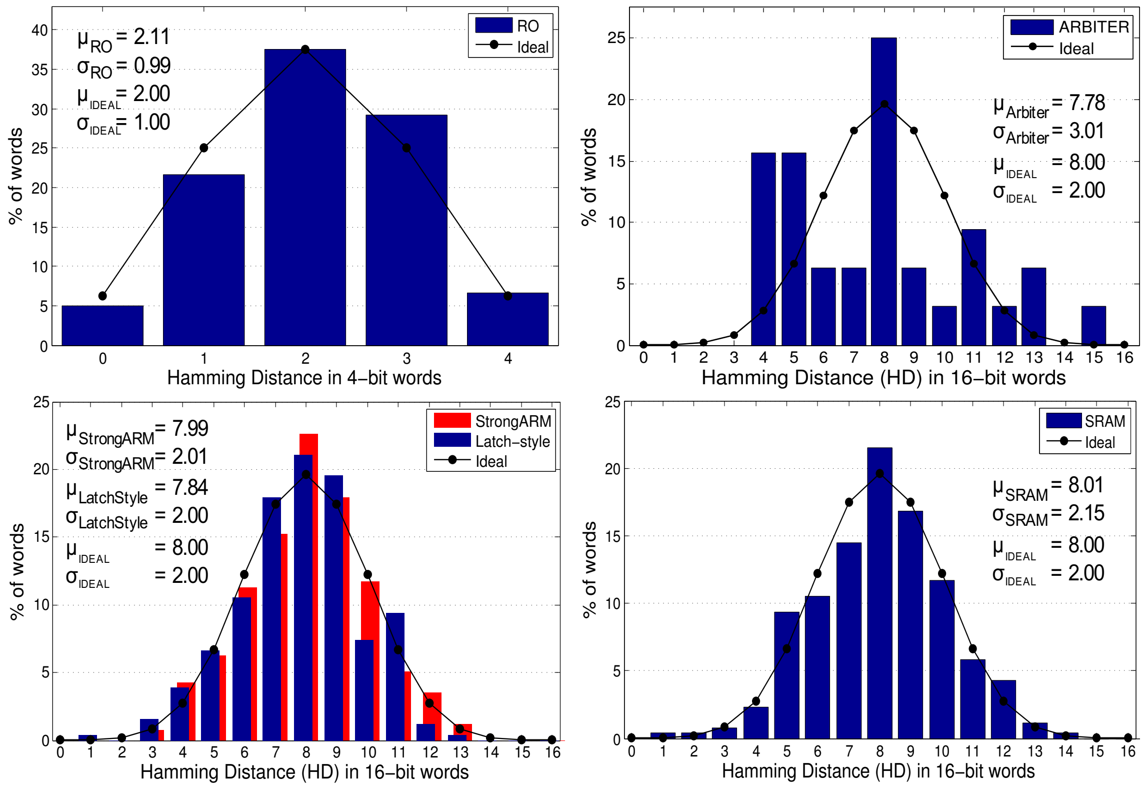


Figure 3.10: Histogram of Hamming distance (HD) of response words across dies for all PUF types. Also shown is the probability mass function of HD in responses from ideally unique dies. The SRAM, SA, and RO PUFs show very close to ideal behavior with mean(μ) and std. dev. (σ) closely matching the ideal behavior. Arbiter PUFs were found to deviate significantly from the ideal behavior which could potentially be because of a bias in their layout.

For measuring randomness, we first ran the NIST statistical tests on the response bits from various PUF types. Next, for the bi-stable PUFs, we measured the percentage of 1's and 0's across the responses across layout orientations and across rows/columns to check for any layout or structure dependent bias. The response of a random, unbiased PUF should have 50% 1's and 50% 0's.

Randomness (NIST tests). To evaluate randomness, we ran the standard National Institute for Science and Technology (NIST) statistical tests [58] on the responses generated from the different PUF types. Note that NIST tests, like other statistical tests for randomness, are designed to test randomness of binary sequences as generated by a random number generator (RNG) under test. Since PUF cores are more like a pool of random bits and do not naturally generate a unique sequence of bits, we generate a sequence of bits from different implementations in the following ways. Arbiter PUFs and ring oscillator PUFs are evaluated using a sequence of random challenges to generate a sequence of bits. SRAM and sense amplifier PUFs are arrayed and a sequence of bits is generated by accessing the arrays in order of the physical location of the cells (first row, followed by the next, and so on). Note that there is no finite set of tests that can guarantee that a sequence under test is truly random. Each test looks for a "pattern", which if detected would indicate that a sequence is non-random.

NIST tests are constructed such that they compare and evaluate a sequence under test to a truly random sequence. The NIST approach involves computing a test statistic (*s*) and its corresponding probability value (P-value). Typically the test statistic is constructed such that large values of a statistic suggest a non-random sequence. The P-value is then the probability of obtaining a test statistic as large or larger than the one observed if the sequence is random. Hence, small values (conventionally, P-values < 0.05 or P-values < 0.01) are interpreted as evidence that a sequence is unlikely to be random. The decision rule in this case states that "for a fixed significance value α, *s* fails the statistical test if its P-value $< \alpha$". For a binary sequence to pass a test, the P-value must be above a significance level (α), which is the probability of Type I error (i.e., the probability of a random sequence failing a test). For example, if the test statistic is constructed such that for an α of 0.001, one would expect 0.1% of all truly random sequences to be rejected by the test and for an α of 0.01, one would expect 1% of all truly random sequences to be rejected by the test.

The results of the NIST tests are interpreted as follows. For example, the tests are run on 1000 binary sequences (i.e., m=1000), each having 5000 bits (i.e., n=5000), the level of significance is 0.01 (i.e., α=0.01), and 996 out of 1000 binary sequences have a P-value $\geq$ 0.01. Statistically, the acceptable range of the % of sequences that pass the test is determined using the confidence interval defined as, 100*$\left\{\hat{p} \pm 3\sqrt{(\hat{p}(1-\hat{p}))/m}\right\}$ where $\hat{p}$ =1-α. This range, for the example above, computes to 99%$\pm$0.94%, i.e. the % of sequences passing (i.e. with P-value $\geq$ 0.01) should be above 98.06%. In our example, 99.6% of sequences pass the test, which is greater than the lower limit of the range and hence the sequences under test are said to have passed the particular NIST test. The conclusion that can be drawn is that the NIST test was unable to find a pattern that would have suggested non-randomness. If a binary sequence passes all the NIST tests, the sequence is "likely" to be a random sequence.

Table 3.1 and Table 3.2 show the result of the application of the NIST test suite on the binary sequences as generated from the various PUF types. The standard NIST test suite has 15 tests, each with its recommended minimum length of the binary sequence under test (n_{min}) for meaningful statistical conclusions. For some of these tests, n_{min} is 10^6. Because of the limited number of samples we could generate from our limited number of testchips, only 7 out of the 15 tests could be run on our data set. For the purpose of testing, m_{total} binary sequences, each of length n_{min} were created from a total of N bits of response from each PUF type (m_{total} = $\lfloor$N/$n_{min}\rfloor$). The level of significance (α) was chosen to be 0.01. The statistically acceptable number of sequences for each test, as per the confidence interval described above, is given by m_{min} and the number of sequences that have their P-value $\geq$ 0.01 is given by m_{pass}. The set of sequences is said to have passed a particular test if $m_{pass} \geq m_{min}$.

Table 3.1 and Table 3.2 show that all PUF types pass all the 7 NIST tests. Also note that while the binary sequence for SRAM and the sense amplifiers are generated by measuring the cells in the array in physical order, the sequences for the Arbiter and ring oscillator PUFs are generated by using a sequence of random challenges (using *randperm* function in MATLAB). The arbiter and ring oscillator PUFs have known vulnerabilities to modeling attacks when used without any post-processing on the challenges [57]. For example, consider an arbiter PUF with

PUF Types ⇒		**Arbiter**				**Ring Oscillator**			
Total Samples (N) ⇒		4704				11873			
		← # Sequences (m) →				← # Sequences (m) →			
Test	n_{min}	m_{total}	m_{min}	m_{pass}	*pass?*	m_{total}	m_{min}	m_{pass}	*pass?*
Freq	100	47	44	46	✔	118	113	118	✔
Block Freq	100	47	44	47	✔	118	113	118	✔
Cusums (f)	100	47	44	46	✔	118	113	118	✔
Cusums (b)	100	47	44	47	✔	118	113	118	✔
Runs	100	47	44	47	✔	118	113	116	✔
Entropy	100	47	44	47	✔	118	113	118	✔
Longest Run	128	36	33	36	✔	92	88	91	✔
Spectral	1000	4	3	4	✔	11	9	11	✔

Table 3.1: NIST randomness test results for delay based PUFs.

PUF Types ⇒		**SRAM**				**SA-S**				**SA-L**			
Total Samples (N) ⇒		28672				8192				8192			
		← # Sequences (m) →				← # Sequences (m) →				← # Sequences (m) →			
Test	n_{min}	m_{total}	m_{min}	m_{pass}	*pass?*	m_{total}	m_{min}	m_{pass}	*pass?*	m_{total}	m_{min}	m_{pass}	*pass?*
Freq	100	286	278	284	✔	81	77	77	✔	81	77	77	✔
Block Freq	100	286	278	282	✔	81	77	81	✔	81	77	80	✔
Cusums (f)	100	286	278	286	✔	81	77	79	✔	81	77	77	✔
Cusums (b)	100	286	278	286	✔	81	77	78	✔	81	77	79	✔
Runs	100	286	278	284	✔	81	77	80	✔	81	77	81	✔
Entropy	100	286	278	286	✔	81	77	80	✔	81	77	80	✔
Longest Run	128	224	217	222	✔	64	60	62	✔	64	60	64	✔
Spectral	1000	28	26	27	✔	8	7	8	✔	8	7	8	✔

Table 3.2: NIST randomness test results for bi-stable based PUFs.

a challenge that is flipped in two consecutive bit locations. The responses in the two cases are very likely to be the same as the two challenges activate and compare nearly the same delay paths. Also, consider a ring oscillator PUF where the raw challenges are decoded to generate the combinations of rings for comparison. A sequence of incremental challenges could decode into a sequence of comparisons of the frequency of several rings with that of a particular ring, which say has a higher than average frequency. Thus, the responses from these sequence of challenges will not be independent and the generated response sequence will be biased. For these reasons, secure implementations of arbiter and ring oscillator PUFs would require some processing of the challenges before they are used internally. To confirm this, we tested responses for the arbiter and ring oscillator PUF when they are activated using a set of unprocessed challenges. Each challenge in this set is generated by incrementing the previous challenge by one. These responses failed all the NIST randomness tests (except *Spectral*), stressing the necessity of randomizing the challenges for the arbiter and the RO PUF for the generation of random responses.

Table 3.1 shows the results for the NIST tests using randomized challenges. All PUF types pass all the randomness tests. Strictly speaking, these results do not conclude that the PUF response are random. Instead, these results conclude that NIST tests were *unable to find any non-randomness* in the response of the PUF types.

Randomness (layout dependent bias). Figure 3.11 shows the bias in the measured response from all PUF types from several testchips. The overall bias (i.e., measured over all bits from a PUF type on a die) across all chips was found to be close to ideal (50%). The measurements for arbiter and RO PUFs are taken across 4 dies, for SRAM across 7 dies, and for SA PUFs across 3 dies (a bonding issue prevented some designs from being tested in all dies).

Since PUF cores derive all their randomness by capturing process variations in their physical implementations, there could be layout or structural dependent patterns, which get masked when we measure the overall bias. For example, a potential source of bias in the array-based PUFs could be the mismatch in the environment of the rows and columns on the periphery as compared to those in the middle of the array. Another potential source of bias could be the layout orientation of the PUF core cell. Hence we measure the bias across rows, columns, and each of the layout orientations for the arrayed bi-stable PUFs to test for systematic biases. Figure 3.12 shows the measured bias across layout orientations for SRAM and both structures of sense amplifiers from multiple testchips. SRAMs and sense amplifiers are laid out in four orientations for area efficiency. These are *R0* (default orientation), *MX* (default mirrored along X-axis), *MY* (default mirrored across Y-axis), and *R180* (default mirrored along both X-axis and Y-axis). The SRAM measurements are taken from 7 chips and the SA measurements are taken from 3 chips. The bias in each orientation and across all chips for SRAM and SA PUFs is shown in Figure 3.13. Figure 3.14 shows the bias computed across row and columns of the SRAM and sense amplifier PUFs. The results of these tests indicate a close to ideally random behavior for all PUF types.

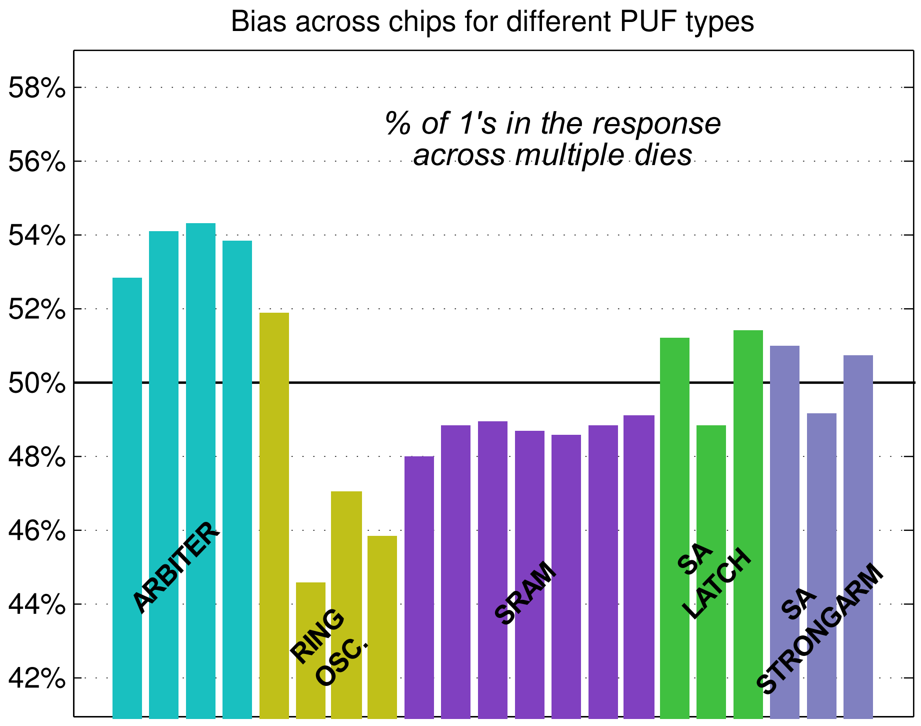


Figure 3.11: Measure of randomness: bias across chips for different PUF types

3.4.3 Reliability

Reliability of the PUF cores was measured as follows.

1. All PUFs cores from a chip were evaluated *once* at 9 different environmental conditions, a combination of supply voltage VDD = 1.2V (nominal) ±200mV and temperature: −20°C, 27°C, 85°C. The temperature was controlled using a temperature controlled cham-

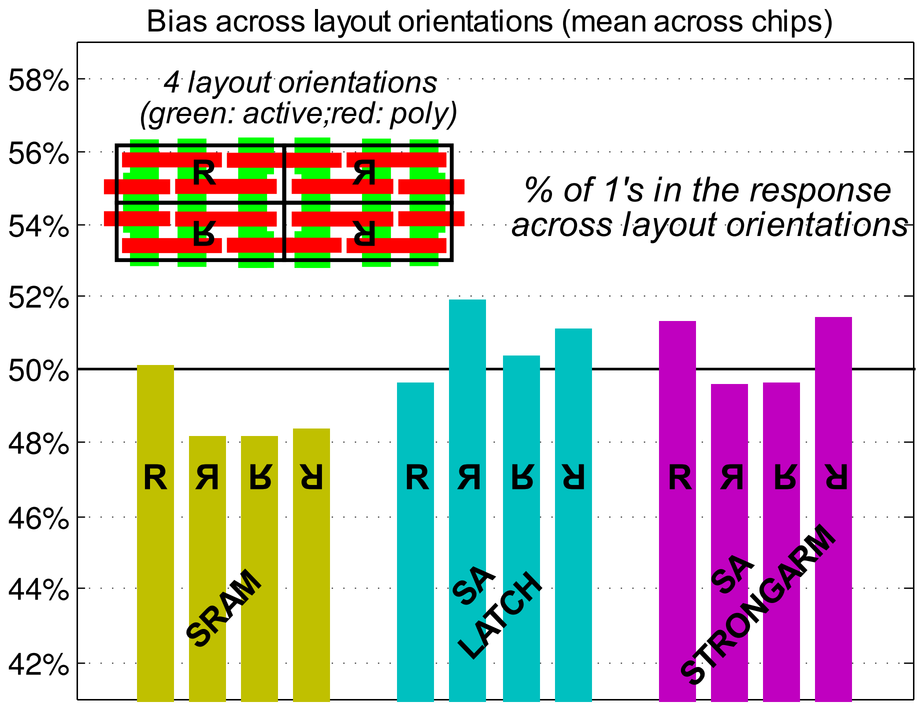
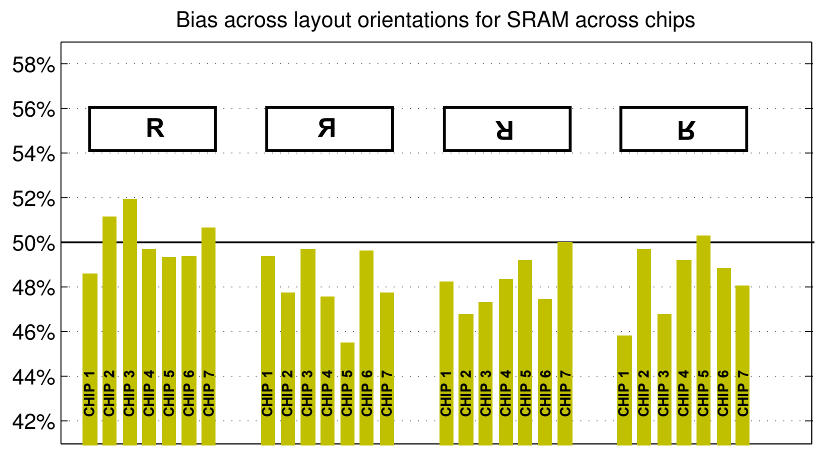
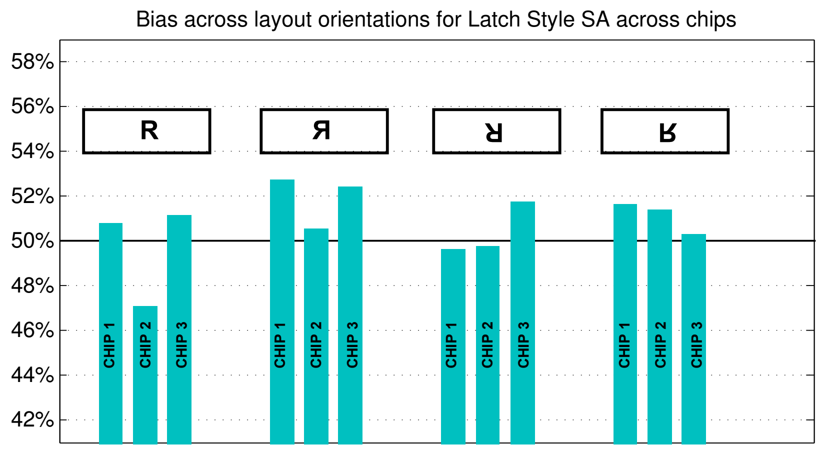


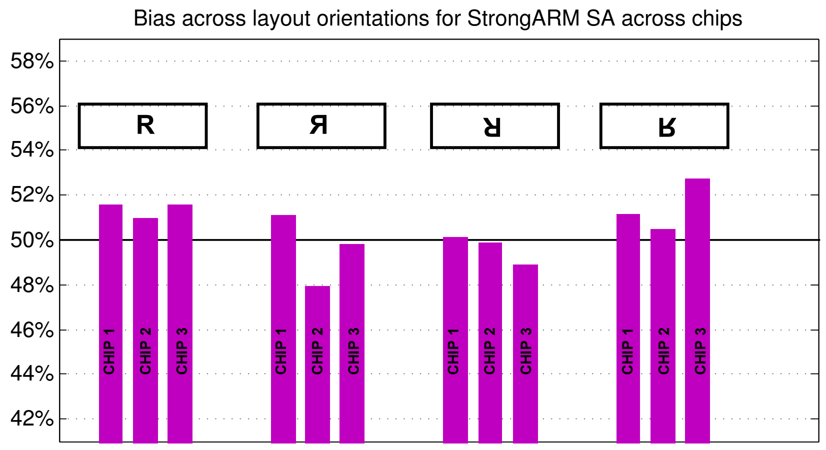
Figure 3.12: Measure of randomness: Mean bias across layout orientations for SRAM and SA PUFs from measurements taken from multiple chips. To compute the mean bias for each orientation, the SRAM measurements are taken from 7 chips while SA measurements are taken from 3 chips.



(a) SRAM: Measured bias from 7 chips



(b) SA (latch style): Measured bias from 3 chips



(c) SA (StrongARM): Measured bias from 3 chips

Figure 3.13: Measure of randomness: Bias across layout orientations for SRAM and SA PUFs across multiple chips. For the SRAM, the bias is measured in 7 chips. For the SAs, the bias is measured in 3 chips.

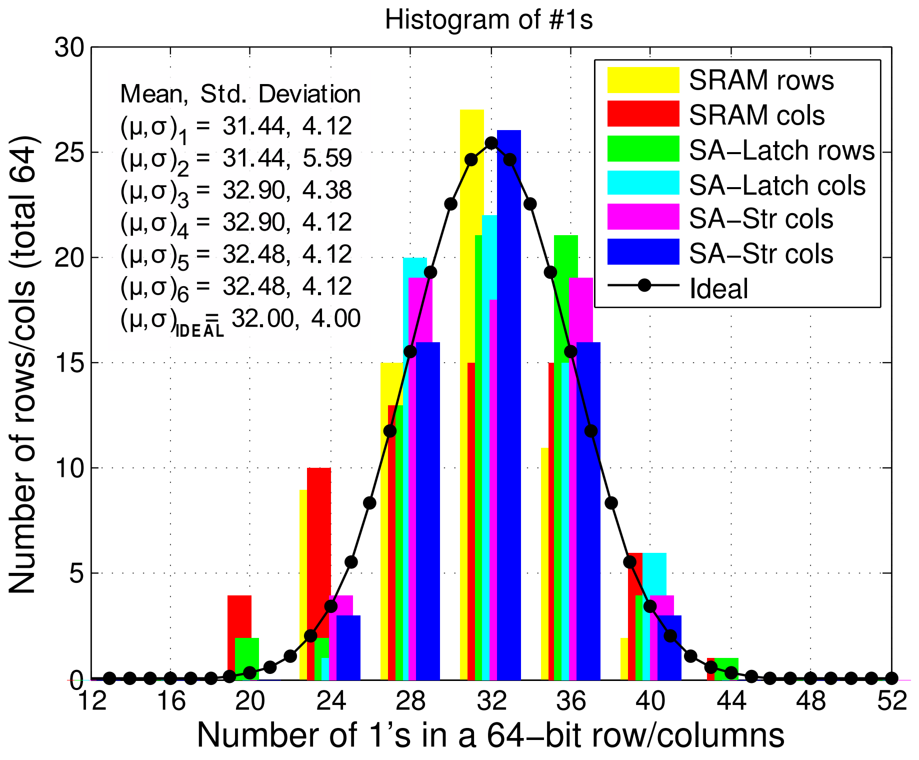


Figure 3.14: Measure of randomness: Histogram of #1s across rows/columns for SRAM and SA PUFs

ber (TestEquity Model 107 [3]). For the SRAM PUF, we consider 4096 bits – one bit for each location of the SRAM array. For the SA PUFs, we have 4096 bits each for the latch style and StrongARM types – one bit each for each location of the SA arrays. For the RO PUF, we measured the frequency of all 32 ROs of type LPsvt (see Figure 3.5 and Table A.2) and generated $\binom{32}{2}$, i.e., 496 bits by comparing the frequencies of all combinations of 2 ROs. For the arbiter PUF, we challenge a arbiter chain with 1000 randomly chosen 64-bit binary strings. The frequencies of the two rings (see Figure 3.2) were compared to generate 1000 response bits for the arbiter PUF.

2. The response bits for each PUF were compared across the 9 conditions and if a bit was found to be inconsistent in even one of the 9 measurements, it was considered an erroneous behavior and the corresponding PUF core was marked as unreliable.
3. The final metric of reliability is expressed in terms of errors and represents the percentage of PUF cores that show an erroneous behavior.
4. We also compute the errors across voltage-only variations (V-only) and across temperature-only variations (T-only) to understand which of the two variations has a stronger impact on reliability.

Figure 3.15 shows the reliability expressed as percentage of errors. For example, an error of 16% (as observed for RO PUFs across V&T variations) implies that 84% of the PUF cores generated response bits that evaluated consistently across all V&T evaluations. Reliability of response of all PUFs was found to be in the range $\sim$73-87% for all PUFs across V&T variations. Further, we find that voltage variations (V-only) have a larger impact on reliability as compared to temperature variations (T-only) for all PUF types except SRAM. For the SRAM, as discussed later in Section 4.2, the power-up values of SRAM does not depend on the voltage to which the VDD is powered-up to. The SRAM bits *lock* to their preferred state much earlier ($\sim$ 200mV) and whether the VDD is ramped up to 1.0V, 1.2V, or 1.4V does not impact the SRAM power-up state in any way. Hence, we do not see any significant V-only errors for the SRAM PUF. However, the SRAM power-up state does depend on the ramp-rate of supply during power-up and thus if the supply voltage is ramped up to different values in the same time, i.e., same time to rise from 0V to the final voltage (and hence resulting in different supply ramp rates), it would impact the

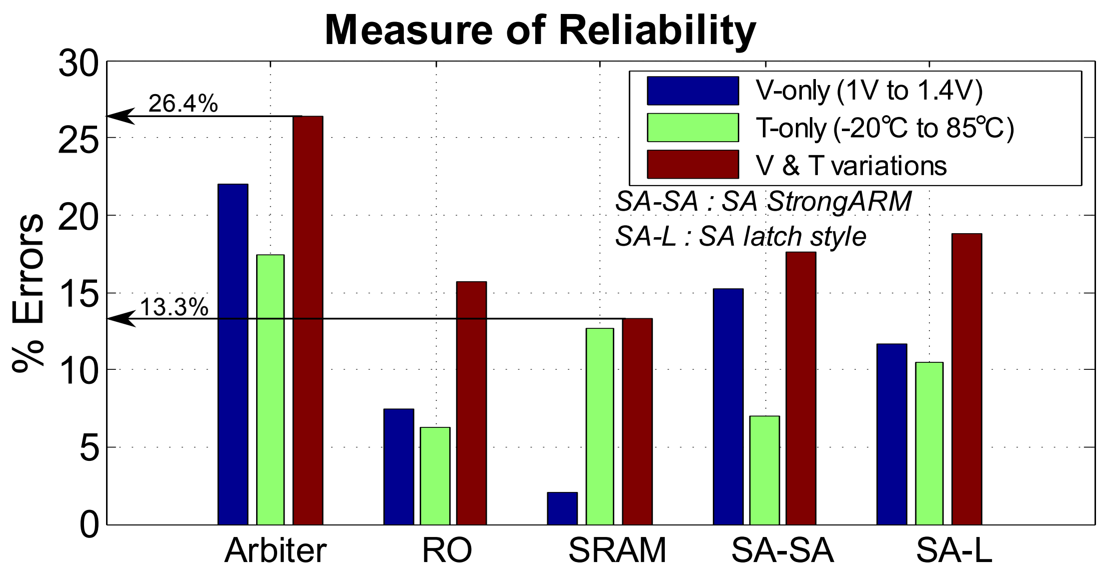


Figure 3.15: Reliability comparison of PUF types across voltage (V-only), temperature (T-only), and voltage and temperature (V&T) variations. The percentage error of all PUF types considered ranges from 13.3% to 26.4%.

reliability of the SRAM PUF.

Note that the measure of reliability as defined above is more conservative than how it is conventionally defined. Conventionally, errors across environmental conditions are computed as the maximum of the errors seen in the response bits, measured at all environmental conditions, and when computed by comparing it to the "golden" reference response (typically measured at the nominal conditions), . Our definition is more conservative and would result in a higher error count. For example, say we had to measure the errors at just at just 2 environmental conditions and we just had 10 bits of response. For the measurements at condition#1, say we see an inconsistent behavior for bit#1 and bit#2 (compared to nominal behavior) and for the measurements at condition#2, we see an inconsistent behavior for bit#2,bit#5, and bit#6. The conventional measure of reliability would count 2 and 3 errors respectively in the two cases and the worst case error count as 3 (or 30% errors) across the 2 environmental conditions. In our definition, we would count the number of bit locations that exhibit an erroneous behavior across any of the conditions. Hence, with our conservative definition, we would count the errors across the conditions to be 4 (one count each for bit#1, bit#2, bit#5, and bit#6), i.e., 40% errors.

The conventional metric of reliability is more suited for applications that would implement ECC for error correction and hence it is important to know the maximum errors seen across any environmental condition to estimate the error correction capability for the ECC design. Our metric of reliability provides a better estimate of the percentage of PUF core bits that are vulnerable to errors across environmental conditions. This number is important to implement the post-silicon selection scheme for reliability enhancement (as discussed later in Section 5).

3.5 Comparison of VLSI Metrics

The baseline VLSI metrics (area, energy, and delay) of the PUFs were evaluated by comparing the cost of generating one response bit.

3.5.1 Area

The area measurements are obtained from the layout of the fabricated design (Figure 3.16. For the arbiter, the area can be amortized by re-using a chain for multiple challenges, but at the cost of increased delay and an increased vulnerability of modeling attacks [57]. The area of SRAM and SA reported is for the 64x64 arrays that include the peripheral circuitry (decoder, I/O, etc.). For a larger array, the effective area of a single SRAM or SA will reduce further. The area of RO includes the area of two rings, and a 25-bit counter to keep track of the frequency of the ROs. RO PUF has an area over 500x higher than a SRAM PUF and over 400x higher than both SA PUF types. Arbiter PUF has an area of over 1300x higher than the SRAM PUF and $\sim$1000x higher than SA PUFs.

3.5.2 Delay

The delays of the arbiter and RO PUFs are measured directly from the chip after on-chip frequency division. The delay of a RO PUF to generate a bit depends on the frequency difference of the selected RO pair. More precisely, a RO PUF response can be sampled only after the differential frequency results in a 1-bit change in the counter values. Assuming a group of ROs oscillating at frequencies that follow a normal distribution (standard deviation = σ_F), it can be

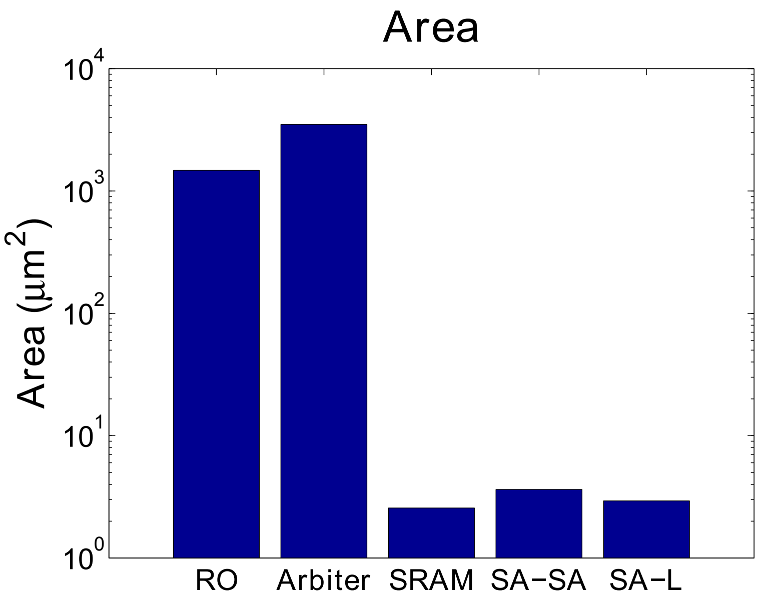


Figure 3.16: Area for the generation of one response bit. Compared PUF implementations are ring oscillator (RO), arbiter, SRAM, StrongARM sense amplifier (SA-SA), and latch style sense amplifier (SA-L). Compared to the area of bi-stable PUFs, the area of RO PUF and arbiter PUF is over 400x and ∼1000x higher respectively. The effective area of the arbiter can be amortized by re-using the chain for multiple challenges, but at the cost of delay and an increased vulnerability to modeling attacks.

shown that when allowed to oscillate for time $T_{99.9} = 555.5/\sigma_F$ or $T_{99} = 55.55/\sigma_F$, 99.9% and 99% ROs, respectively, can be expected to have at least 1 bit difference in the counter value. The oscillation frequency of rings was measured from chip to compute σ_F. The delay reported for RO in Figure 3.17 corresponds to $T_{99.9}$, Due to test setup limitations, the SRAM and SA arrays could not be tested at-speed for access time measurements. Their reported delays are based on simulation of post-layout extracted netlists. They were, however, tested for functionality at a lower clock speed of 10MHz making them still much faster than RO PUF and comparable to the arbiter PUF delays. As seen in Figure 3.17, RO PUF has a delay of $\sim 7.5*10^4$ higher than the bi-stable PUFs and the arbiter PUF has a delay of over 10x higher than the bi-stable PUFs.

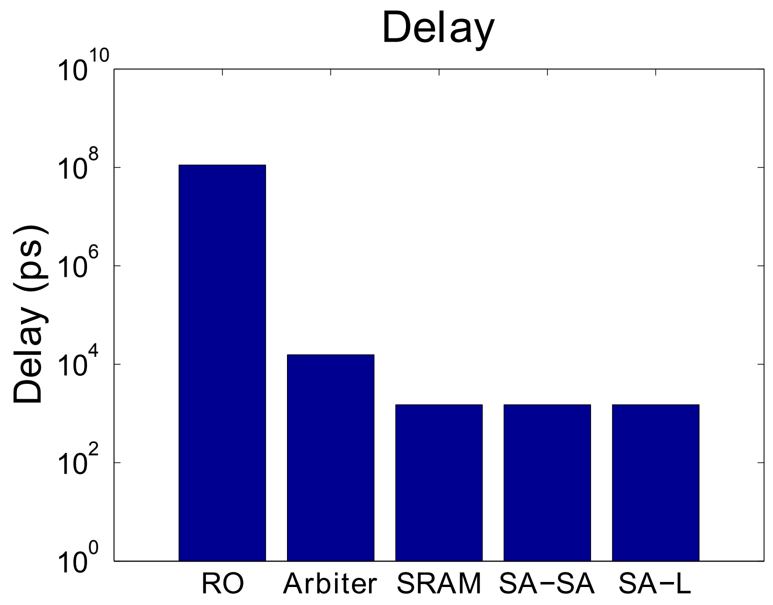


Figure 3.17: Delay for the generation of one response bit. Compared PUF implementations are ring oscillator (RO), arbiter, SRAM, StrongARM sense amplifier (SA-SA), and latch style sense amplifier (SA-L). Compared to the delay of bi-stable PUFs, the delay of RO PUF ($T_{99.9}$) and arbiter PUF is $7.5*10^4$x and $\sim$10x higher respectively.

3.5.3 Energy

The energy results for the PUFs are generated from similar simulations of post-layout extracted netlists (Figure 3.18). The RO and arbiter PUFs have $\sim 10^5$x and $\sim$8x higher energy consumption as compared to the bi-stable PUFs.

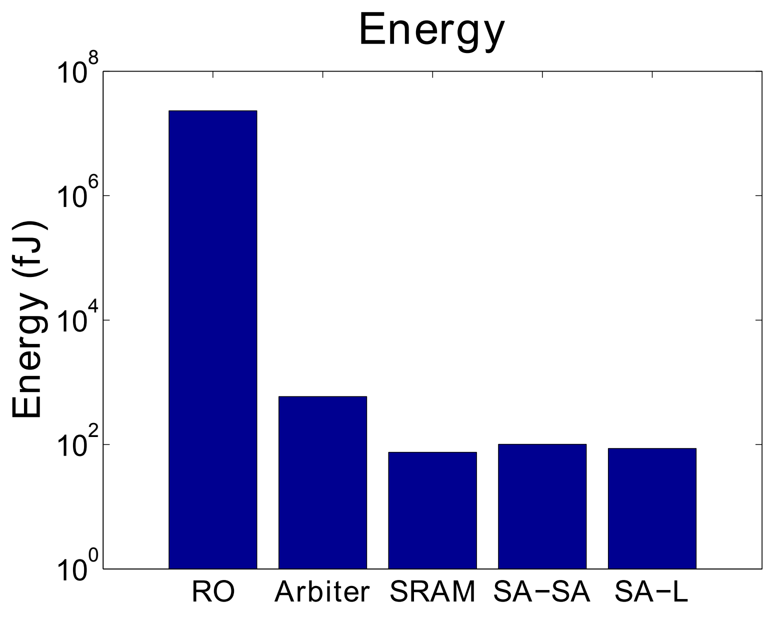


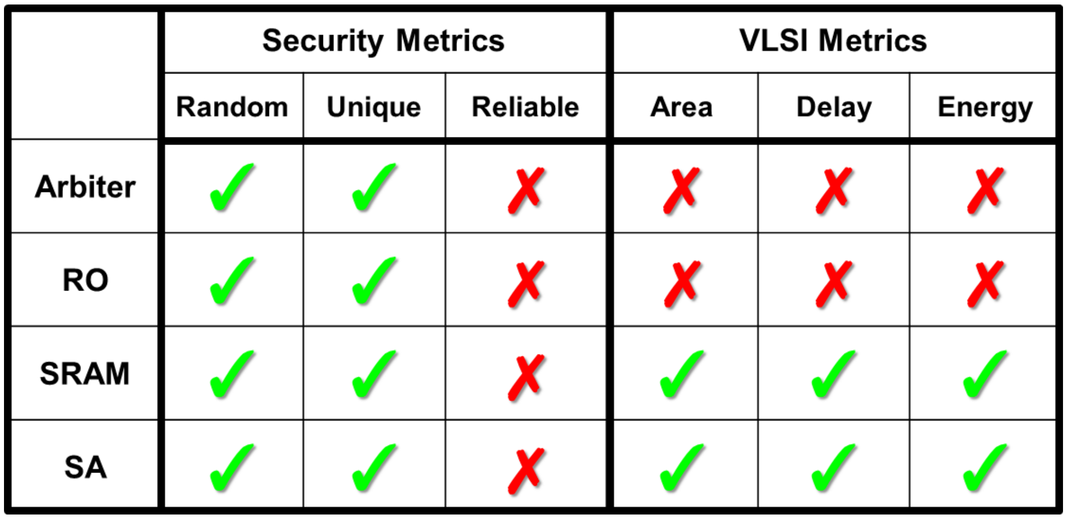
Figure 3.18: Energy for the generation of one response bit. Compared PUF implementations are ring oscillator (RO), arbiter, SRAM, StrongARM sense amplifier (SA-SA), and latch style sense amplifier (SA-L). Compared to the energy consumption of bi-stable PUFs, the energy consumed by RO PUF and arbiter PUF is $\sim 10^5$x and $\sim$8x respectively.

3.6 Summary of Comparison

From the measurements, we can conclude that all PUF cores, when designed carefully, generate response bits that have good randomness and uniqueness properties. However, the reliability of the raw PUF response bits is insufficient for direct use in applications that require high/perfect reliability. The errors in responses for various PUF implementations were found to be in the range 13-26% when responses were measured across $\pm$200mV VDD variations and across a temperature range of -20°C to 85°C. We also conclude from the comparison that the bi-stable PUFs have superior VLSI metrics. These conclusions are summarized in Figure 3.19.

3.7 Reliability, a Hard Problem

In the remainder of this chapter, we discuss techniques to achieve high reliability in PUFs. First we provide a brief overview of the conventional technique of using error correction codes (ECC)



	Security Metrics			VLSI Metrics		
	Random	Unique	Reliable	Area	Delay	Energy
Arbiter	✓	✓	✗	✗	✗	✗
RO	✓	✓	✗	✗	✗	✗
SRAM	✓	✓	✗	✓	✓	✓
SA	✓	✓	✗	✓	✓	✓

Figure 3.19: PUF comparison summary

that *correct* the errors in the PUF response. It turns out that the ECC techniques can have high overheads that grow quickly with higher error correction capabilities. Also, several studies have indicated that many of the ECC techniques have security vulnerabilities that require further investigation. After a brief overhead analysis of the ECC techniques, we describe several low-overhead error *reduction* techniques. In Section 3.7.2, we categorize these error reduction techniques as either **extrinsic** or **intrinsic**.

3.7.1 Conventional Error Correction

The conventional method to improve PUF reliability is to use powerful error correction codes (ECC) to correct the raw response from the PUF core. In [14], the authors describe a theoretical framework using code-offset based syndrome generation (fuzzy extractors) for error correction in a noisy response. Key extraction from a SRAM PUF using linear block codes was described in [20]. In [11], the authors use concatenated block codes for improved hardware efficiency and correction capability. In [40, 44], the authors propose the use of conventional soft-decision decoder and show its coding gains over their hard-decision counterparts. This, however, results in additional evaluation time and computational steps and a significant increase in the size of helper data. An alternate soft-decision decoder that requires only one evaluation and a smaller helper data footprint was proposed in [37]. All these error correction implementations require careful design as the helper data that is generated for error correction (typically assumed to be publically stored or transmitted in the clear) can be a potential source of information leakage. This vulnerability is discussed in [73] where the authors also propose an alternate and more secure technique for error correction which was further improved in [74]. Yet another alternate method for error correction based on pattern matching was proposed in [53].

Unfortunately, these error correction implementations have significant VLSI overheads. For instance, and assuming a bit error rate (ber) = 0.15, typical implementations would require $\sim$3000–10000 raw response bits to generate a reliable 128-bit key, i.e., $\sim$20-80 bits for the generation of a single reliable bit [37]. Further, there is an additional area overhead because of the correction logic circuitry. Note that [37] and many of the other implementations have assumed a secrecy rate of $\sim$0.75 (first used in [18]) and hence their reported overheads are for the generation of 171, i.e., $\lceil 128/0.75 \rceil$ bits which is later compressed (privacy amplification) to 128-bits of perfect entropy. Privacy amplification amounts to applying a universal hash function and is not a focus of our work.

The cost of error correction scales up very quickly with correction capability. For example the BCH coding in [18] requires $\sim$26.7 raw response bits for the generation of a single reliable bit if the ber = 0.15 and requires only $\sim$3.68 raw response bits if ber was reduced to 0.06.

Another concern with most ECC implementations is that the size of helper data generated is proportional to the size of the raw response bits required. This helper data needs to be stored on a non-volatile memory (NVM) along with PUF or transmitted in the clear, adding to the NVM storage costs or the transmission costs. For the hard-decoder ECC implementations, the size of helper data is close to the size of the raw response bits required. But for the soft-decoder implementations, the size of helper data can be significantly larger (e.g., $\sim$10-20x of the size of raw response bits required) depending on the amount of soft-information stored per bit [44].

This helper data needs to be loaded on the die at the time of decode. This can be done in two ways. One way is to load the entire helper data to an on-die memory, say a SRAM. Hence, the decoder can quickly access the words of the helper data at a high speed ($\sim$ access time of SRAM). However, this would require a SRAM on die equal to the size of the helper data, which is significant, as we discussed above. The other way is to load only a few words at a time on-die, say to a register file. This will minimize the on-die memory requirements but would limit the speed at which the decoder can be clocked. This is because the register file has to be updated with more words of helper data from the off-chip NVM and therefore the decoder speed will be limited by the access time of the off-chip NVM, which can be a few orders of magnitude higher than a typical on-chip memory access time.

Another concern is the amount of entropy left in the PUF bits after encoding. An argument against the typical code-offset based ECC implementations was made in [73] where it is shown that if a 33x repetition code is used with a PUF that has a bias of 0.5152, then statistically, every single bit is leaked out, i.e., no entropy is left in the generated bits after decoding. Another example is the key generator implementation presented in [42]. They extract bits from the frequency ordering of ROs in a RO PUF as was first proposed by [71]. They assume that the secrecy rate (i.e., amount of entropy per bit) in the encoded bits is 97.95%. This comes from the ideal case assumption that all RO frequencies are i.i.d.'s. Their implementation generates 2226 bits which is equivalent to 2226*97.95/100 = 2180.4 bits of entropy. Their implementation of helper data leaks 2052 bits of information and hence the the remaining entropy in the decoded bits = 2180.4-2052 = 128.4 bits. However, if there is any systematic bias in the ROs that result in a non-uniform frequency distribution, the secrecy rate will be lower. If the secrecy rate is even 6% off from the assumed ideal case assumed, there would be no entropy left in the generated bits.

Finally, these implementations take a large number of cycles to decode the bits. For example, the implementation in [11] takes 25-30k cycles to generate 128 reliable bits and the key generator implementation in [42] takes $\sim$300k cycles to generate a 128-bit key.

To summarize, the concerns with typical ECC schemes are as follows.

- Cost of ECC scales quickly with error correction capabilities.
- Almost 20-80 source bits required per bit of reliable bit generation.
- Large helper data results in large off-chip NVM storage or increased transmission costs.
- Depending on the loading scheme of helper data on die during decode, either a large on-die memory is required or the decoder speed is significantly bottlenecked.

- Entropy left in the decoded bits is very sensitive to the ECC scheme and the randomness of the source bits.
- Decoding requires a lot of cycles.

Because of these reasons, it is desirable if raw bits from a PUF have high baseline reliability. Hence, researchers have proposed various techniques to reduce the number of errors in the baseline PUF response. These schemes are introduced next in this chapter and our work on these techniques is presented in the next 3 chapters.

3.7.2 Error Reduction Techniques

Researchers have proposed several techniques to reduce the errors in the raw PUF response. Most of these reliability enhancement techniques for error reduction can be categorized as either **extrinsic** or **intrinsic**. Extrinsic techniques only leverage the external control to the PUF core and the fundamental structure of the PUF core remains unchanged. In general, extrinsic techniques are easier to implement and can reduce errors by 70-80%. Intrinsic techniques, on the other hand make more fundamental changes to the set of PUF core bits. They either involve making changes to the PUF core devices or to the selection criteria of bits from a set of PUF core bits. Their implementations have slightly higher overheads compared to extrinsic implementations, but are able to achieve $\sim$100% reduction of errors requiring very low or no ECC implementation to achieve a desired level of reliability.

3.8 Summary of Chapter

In this chapter, we have presented an apples-to-apples comparison of several PUF implementations using measurements from test-structures built in a 65nm bulk CMOS testchip. The comparison results are summarized in Section 3.6. Measurements show that while uniqueness and randomness is achievable, reliability of the raw PUF response is insufficient for applications that require perfect reliability. We then discuss the techniques to achieve high reliability. We provide an overview of the conventional techniques of error *correction*, along with a discussion of the associated overheads. Alternately, there are several error *reduction* techniques that can minimize or completely eliminate the use of high-overhead ECC techniques. These techniques can be categorized as either *extrinsic* or *intrinsic*. In the next chapter, we discuss our work on extrinsic error correction techniques, followed by a discussion on our work on intrinsic techniques in Chapter 5 and Chapter 6.

Chapter 4

Extrinsic Techniques for Reliability Enhancement

In the previous chapter, we provided an analysis of security metrics of various PUF implementations. We saw that achieving high reliability in the raw response is a challenge for most PUF implementations. Conventional method to achieve high reliability is to correct the errors in the raw PUF response using error correction codes (ECC). As discussed in the previous chapter, the overheads associated with ECC can grow quickly with error correction capabilities. Several error *reduction* techniques exist that can minimize the use of error *correction* techniques. These error reduction techniques (or reliability enhancement techniques) have been categorized as either *extrinsic* or *intrinsic*. In this chapter, we will discuss our work on the extrinsic reliability enhancement techniques. We categorize a reliability enhancement technique as extrinsic if it involves no change to the fundamental structure of the PUF core. It only leverages the external control signals to the PUF to improve its reliability. We study the following two extrinsic reliability enhancement techniques - multiple evaluation (ME) and activation control (AC).

4.1 Multiple Evaluation (ME)

4.1.1 Introduction

The response of a particular PUF instance each time it is evaluated has a finite probability of being different, even at the same environmental conditions, due to random noise in the system (e.g., Johnson-Nyquist noise, random telegraph noise, uncorrelated crosstalk). One way to increase the reliability is to evaluate the PUF multiple times, and then combining the "soft" responses to determine the final "hard" binary response. In this section, we evaluate the efficacy of such a *Multiple Evaluation (ME)* scheme for increasing PUF reliability. This technique, however, is limited to canceling noise and cannot correct errors that are due to any nominal shift in performance at a different operating environment or due to aging of devices. In this work, we study the error reduction capabilities of ME at the nominal conditions.

4.1.2 Delay-based PUFs

Due to the inherent nature of the ring-oscillator PUF and the nature of the modified test-structures for the arbiter PUF on our testchip (which allows the delay chains to oscillate), there in an inherent averaging of delays over multiple cycles. Hence, ME is not expected to result in any significant reduction in errors for RO PUFs. We ran 50 frequency measurements on all the 256 ring-oscillators on chip and the maximum variation in frequency (i.e., difference in the maximum and the minimum measured frequency) for each of the 256 RO was less than 0.2% of its mean frequency. These variations could be due to temporal fluctuations in the VDD, which would then have a similar impact on the other ROs in a typical RO PUF implementation (Figure 2.6). Hence, such frequency fluctuations may not contribute to errors in the response bits from a RO PUF. As a comparison, the standard deviation of frequency of LPsvt and GPsvt ROs as a percentage of their means was found to be $\sim$1% and $\sim$1.4% respectively (see Figure 3.6). To summarize, we can say that ME is not an efficient technique to reduce errors in a RO PUF.

4.1.3 Bi-stable PUFs

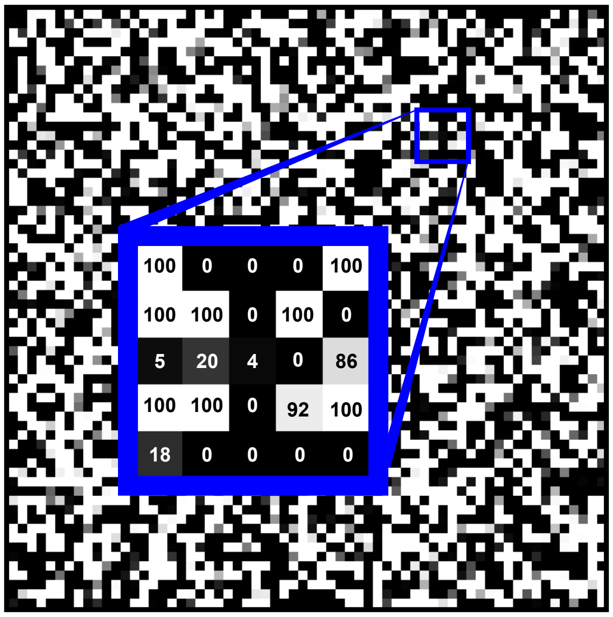
Figure 4.1 shows the evaluation maps of a 64x64 SRAM PUF array (1000 power-ups) and a 64x64 SA PUF array (10 enables) from one testchip. The "soft" evaluation data is represented by the greyscale coloring of each square indicating the number of times the site evaluated to 1 versus 0.

Generating the Golden Map. At time 0 manufacturing testing of a PUF, it is often desirable to dump a "golden" evaluation map of the PUF core random bits as a compact way to store the challenge-response behavior, rather than generate and store the entire (huge) CRP database [32]. This is typically done via some form of "backdoor" access path that is disabled (e.g., via e-fuse, laser, or FIB) before the product is deployed. By using multiple evaluations, we can ensure that the golden evaluation map is as representative as possible of the highest probability PUF behavior.

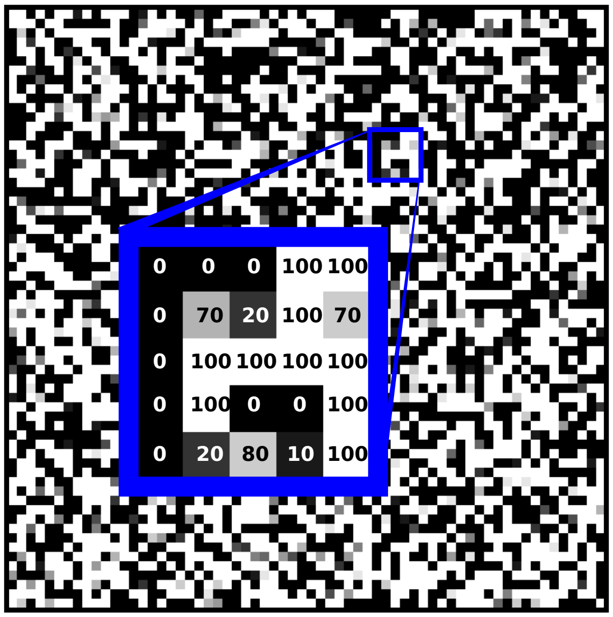
In Figure 4.2 we compare the initial evaluation map generated with N evaluations and compare it with a golden evaluation map as generated from 1000 power-ups. The average one-time evaluated evaluation map differs $\sim$2-2.5% from the golden bitmap. To get errors below 1%, $\sim$7-9 power-ups are required, and for below 0.5%, $\sim$30 power-ups are required. Also, the % errors go down to 0 as expected when power-ups approach 999, since it is what it's compared against for error calculations. This measured data is gathered from 7 testchips.

Number of In-the-Field Evaluations. In a ME scheme, one of the key design decisions is how many evaluations are needed to generate a reliable final response. Figure 4.3 shows how the percent error decreases with the number of evaluations for an SRAM PUF using a majority vote combining stage for 7 different 65nm testchips. The assumed correct "golden" response is generated using 100 evaluations performed separately. It was seen that for 10 evaluations, errors were below 1%, and for 40 evaluations, errors were below 0.5%.

Implementation Overheads Multiple evaluations in-the-field will have overhead costs in area, delay, and energy. Designing a PUF for N evaluations in-the-field will require at least N times the evaluation time, not counting any fixed overheads for start-up. For SRAM PUFs, once powered up, the response generation time for multiple challenges is reasonably small (typically $<$10 ns).



(**a**) SRAM array



(**b**) SA array

Figure 4.1: Greyscale bitmap showing the power-up values of a 64x64 SRAM array (a) and latch-style SA array (b) as generated from 1000 power-up/10 enable measurements at 27°C and nominal V$_{DD}$ of 1.2V. A zoomed in section of the array is shown with percentage of times the element resolved to '1'. In the bitmap, a white pixel represents an element always resolving to a '1', and black pixel represents an element always resolving to a '0'. A grey pixel represents an element that sometime resolves to '1' and sometimes to '0' with the relative ratio indicated by greyscale value.

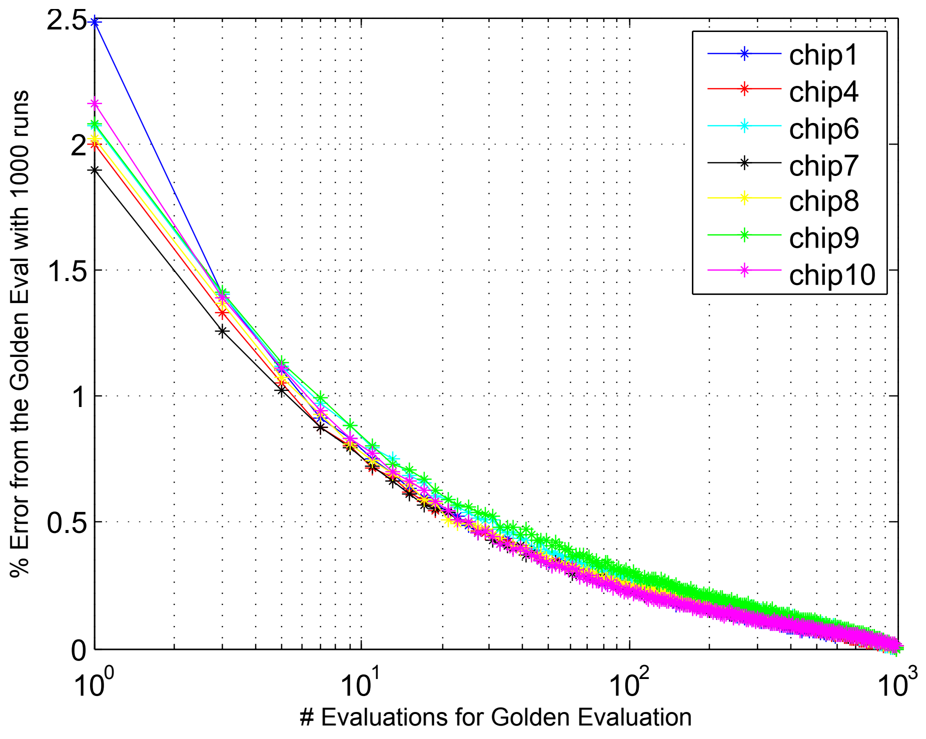


Figure 4.2: Percentage errors in initial response bitmap generation when majority voting done after N evaluations (1 to 999). Errors are calculated against a bitmap generated from 999 power-ups followed by majority voting. The data shown is for 7 different chips. The one-time evaluated bitmap differs by ~2-2.5% on average from that generated from 1000 power-ups for all chips. Further, to get the error below 1%, around 7-9 power-ups are required, and for error below 0.5%, around 30 power-ups are required. % errors go down to 0 as expected when power-ups reach

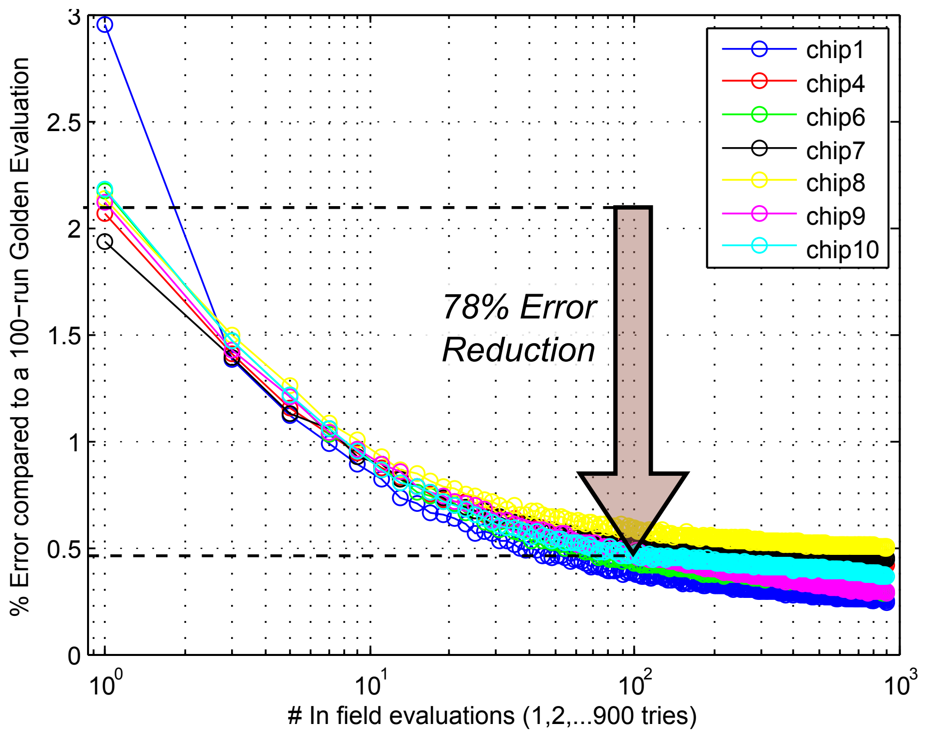


Figure 4.3: Percent errors in response bitmap generated in-the-field when majority voting done after N evaluations. N ranges from 1 to 899 and errors are calculated against a golden response bitmap generated from 100 power-ups followed by majority voting. The data shown for 7 different chips. The average one-time evaluated bitmap differs ∼2-2.5% from the golden response. Further, to get the error below 1%, around 7-9 power-ups are required, and for error below 0.5%, around 40 power-ups are required. Also, the percent errors improvement is significantly slower after the initial few evaluations.

However, multiple evaluations would necessitate multiple power-ups each of which can take up to 1 ms each, since after each power-down we need to wait for the previously stored state to discharge. The SRAM data retention time was 1 ms at 27°C, 100 ms at -40°C, and 0.04 ms at 85°C from measurement of the testchip. SA PUFs can have a faster evaluation time (typically $\sim$1-10 ns), since smaller sections of the design are being activated versus an entire block power-up/down. Note that a SRAM PUF could be re-designed such that only the SRAM cell array or a portion of the array is powered-up/down per evaluation and could achieve low evaluation times similar to the SA PUF, but current SRAM PUFs that use generic compiled SRAM macro, not specifically designed for PUF use, do not have this capability.

Another key design decision is the type of evaluation combining function to use. We can simply use a 50% threshold majority function. Or the voting threshold could be adjusted to compensate for any systematic skew in the evaluation value distribution. For example, if the cells systematically skewed more towards evaluating to 1 due to a layout asymmetry, the threshold could be adjusted up to compensate.

A majority function can be built using an up-down counter to count the resolved polarity of a bit for every evaluation. An N-bit counter, with non-saturating counting, can at most count for $2^N - 1$ evaluations. Any more evaluations would require a larger counter. These overheads in area and/or delay could offset some of the area and delay advantages of bi-stable PUFs over the delay PUFs. However, the overhead for a majority function for in-the-field multiple evaluation may be significant, since it requires storage of all the evaluation values in counter registers for each bi-stable element and thus would require at a minimum $2 \cdot log_2 N$ bits to store.

The PUF could be evaluated in K sections, thus reducing the storage overhead by roughly K, but this would also increase the delay by a factor of K. Additionally, we could use saturating counters and store fewer bits, but at the cost of loss of accuracy in evaluating the majority vote. Finally, we note that while ME could also be used with delay-based PUFs, it is best suited for bi-stable PUF designs, due to their short evaluation times, rather than delay-based PUFs which require comparatively longer evaluation times.

4.2 Activation Control (AC)

4.2.1 Introduction

Activation control refers to controlling external signals (e.g., voltage level, slope) during the activation of a PUF with the aim of creating conditions that result in minimum errors in PUF response. In [69], propose that lowering supply voltage and tuning of body bias can result in a significant increase in reliability of ring oscillator PUFs. Since the results are based on simulations (in 90nm technology) and a very limited number of Monte Carlo runs (only 20), the claim of 100% reliability needs further analysis.

SRAM PUFs are typically activated by ramping up the VDD power supply after resetting the internal nodes to ground. It is known that the reliability SRAM PUF is independent of the value of VDD [59], but depends on the VDD power-on ramp [26, 60]. Most studies suggest that a slower ramp rate will result in increased errors in SRAM PUF. The results we arrive at are, at least at the face value, different from this conclusion. However, a deeper analysis shows that this conclusion is a special case for which our results are consistent with those in the literature. In this section, we provide an extensive evaluation of the effects of VDD ramp rate on the reliability of the SRAM PUF using 3 different voltage sources. The results are based on measurements on the SRAM PUF test-structures on our Generation I testchip (Section 3.3.1).

We performed Monte Carlo simulations on a SRAM cell using 65nm models. We discharged the cell to ground and then provided a ramp of $100\mu s$. We found that most cells have already resolved, as can be seen by the voltage differential generated in the internal nodes, by the time the VDD has risen to $\sim$200mV (Figure 4.4). This was also confirmed from our measurements, as well as results already published in literature, as discussed earlier.

4.2.2 SRAM PUFs: VDD Power-up Ramp Control

Figure 4.5 shows the average percent errors across 1000 power-ups for different VDD ramp rates when swept from 0V to 1.2V. For these error calculations, the golden bitmap for reference was generated by majority voting from the 1000 power-up measurements at the particular ramp rate. The figure shows a steady decrease in the percentage of errors, from 3.5% to 0.67%, with an

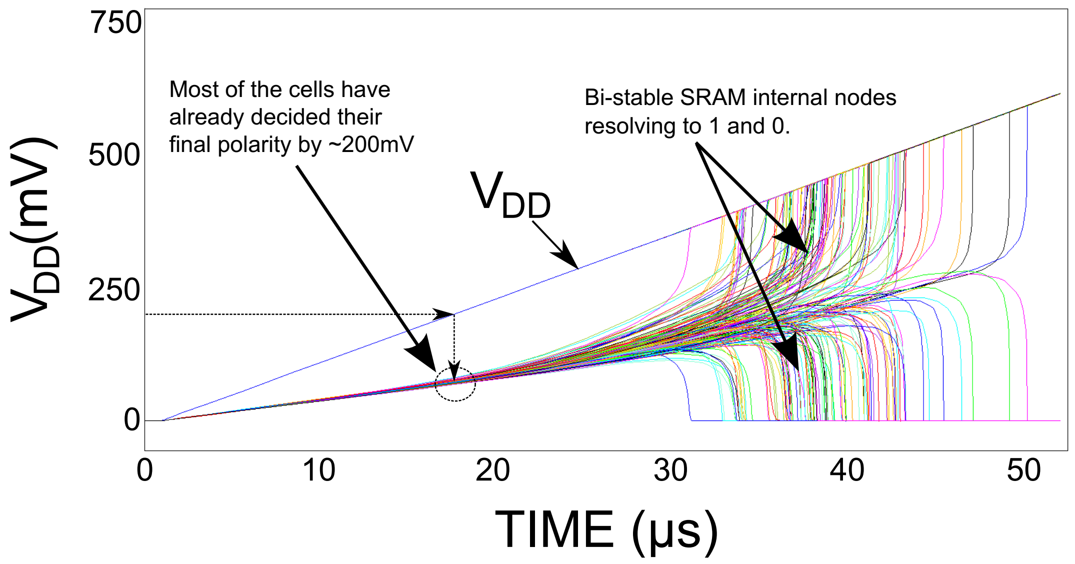


Figure 4.4: Waveform showing internal nodes of a SRAM with VDD ramp rate of 100μs from 0-1.2V for a 1000 run Monte Carlo simulation in 65nm. As seen, most of the decisions of bit polarity are taken at quite a low voltage of VDD. Simulations show that most internal nodes have already started generating a voltage difference, indicating a move from the meta-stable state to one of the stable states, at VDD as low as 150-200mV.

increase in the ramp rate from 0.8ms to 14s. We used three different sources of ramp generation (with different range capabilities) and found a similar trend for all the sources. For two of the sources, we used external resistors in series with the power supply to reduce the ramp rate. This resistor, along with the on-chip parasitic capacitance of VDD, acts as a low-pass RC filter and potentially reduces the high frequency noise in the supply voltage to the SRAM. Figure 4.5 shows a significant reduction in errors for measurements with resistance in series, suggesting that a lower supply noise results in a more reliable PUF. This analysis can assist a designer in choosing a ramp rate when designing in a system that generates VDD ramp rate using internal circuitry.

Figure 4.6 also shows percentage error results where instead of generating the golden bitmap at every ramp rate, this reference is generated at a particular chosen ramp rate. This analysis is pertinent in cases where the PUF is characterized at a particular ramp rate, but is evaluated in field (maliciously or otherwise) at different ramp rates. For Figure 4.6(a), the reference bitmap is generated for our highest ramp rate point of 0.8ms. The percentage errors for all lower ramp rates in this case is higher than that at the reference point, which is consistent with previously

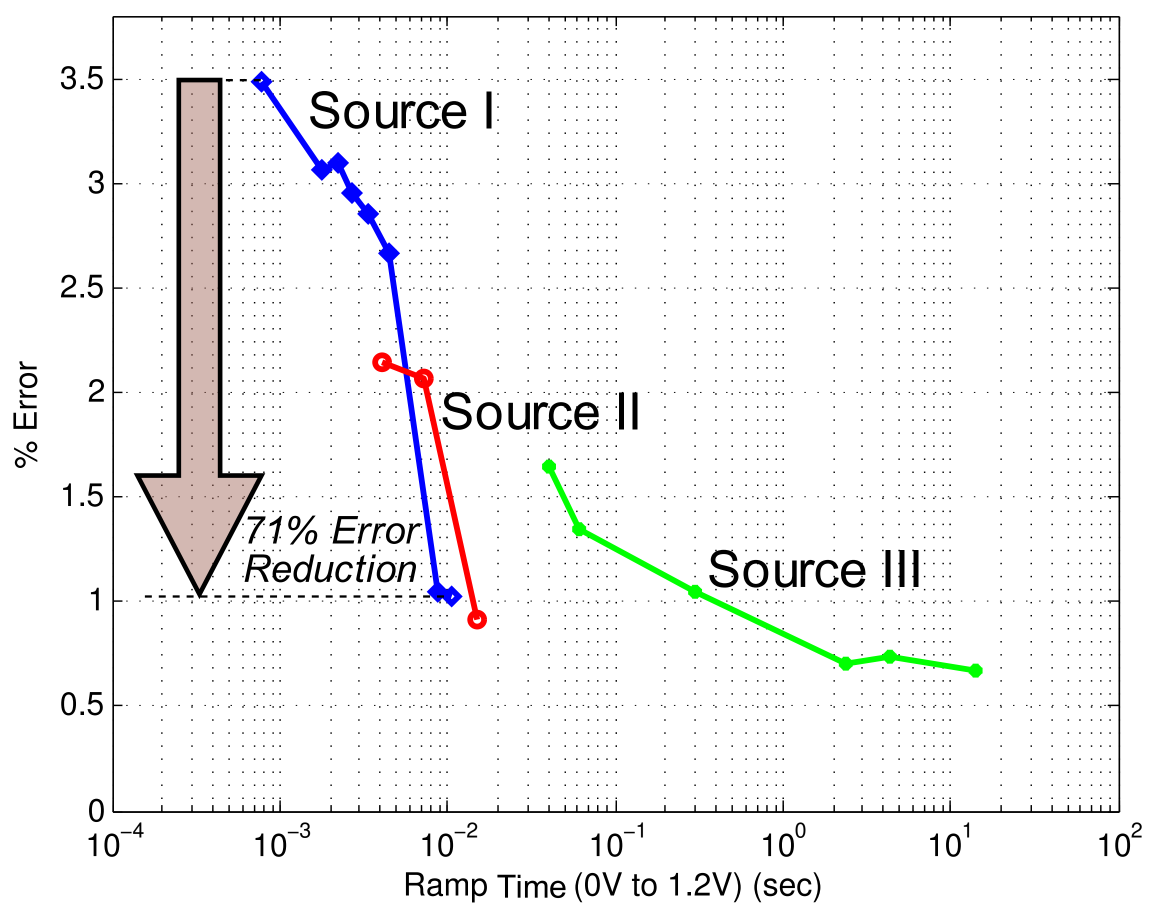


Figure 4.5: Reliability of a SRAM array when the golden reference bitmap is also generated at the same ramp rate as the evaluated array. The % errors at every ramp rate were computed as an average over 1000 SRAM power-up measurements at 27°C. We used three different sources of ramp generation (with different range capabilities and noise characteristics). A steady decrease in the percentage of errors, from 3.5% to 0.67%, was seen when the ramp rate was swept (0V to 1.2V) from 0.8 ms to 14 s.

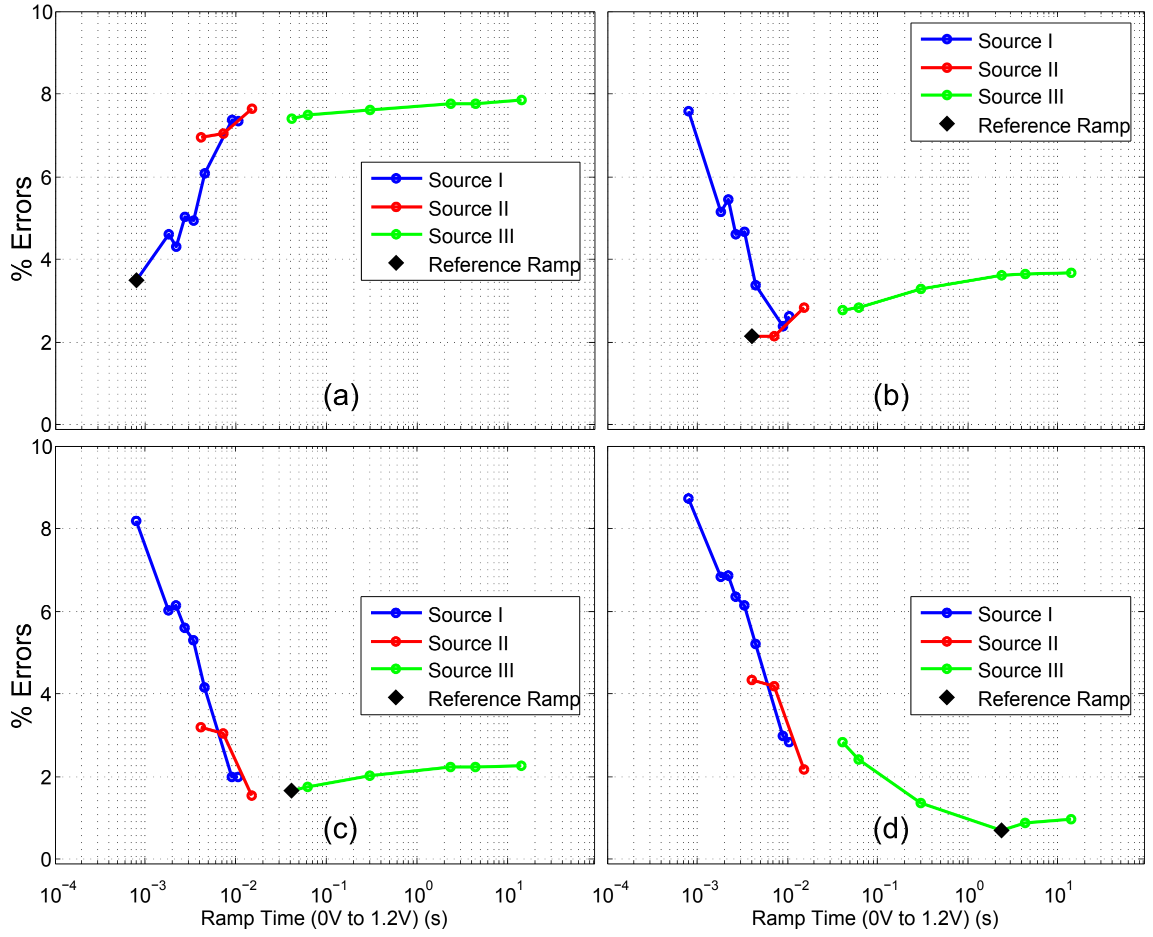


Figure 4.6: Reliability of a SRAM array across various ramp rates but when the reference bitmap is generated at a given ramp rate. (a) shows the average percentage errors for 1000 power-up measurements taken at 27°C at various ramps and when compared against a golden bitmap generated by a majority voting of 1000 power-ups at the reference ramp rate of 0.8 ms. (b)-(d) show similar results for other reference ramp rates.

reported reliability data [60]. However, if the reference bitmap was generated at an intermediate ramp rate, as shown in Figure 4.6(d), the percentage errors at all ramps, with both higher and lower rates, were higher than that for the reference ramp. Moreover, larger the difference in the ramp rate from the reference ramp, larger were the observed percentage errors.

Summarizing the results from Figure 4.5 and Figure 4.6, we can conclude the following to achieve high reliability.

1. The VDD ramp rate for in-field evaluation should be equal or as close as possible to the ramp rate that was used for the evaluation of the golden bitmap (Figure 4.6). For example,

if the golden bitmap is generated at 100ms, then in-field evaluation at ramp times both $>$100ms or $<$100ms will have higher errors compared to the case when the ramp time is exactly 100ms. Also, errors seem to be higher for the case when in-field ramp time is smaller (faster ramp) compared to the golden ramp time.

2. If it can ensured that the ramp rate at in-field evaluation is consistent with that during golden bitmap evaluation, then the ramp time should be chosen as high as possible (Figure 4.5).
3. A less noisy source results in lower errors (Figure 4.5).

A possible explanation of these results is that the SRAM power-up state is function of the relative strengths of the devices in the cross-couple as well as any differential coupling capacitance of the internal nodes with the VDD. A slower ramp rate results in slower charge injection and a faster rate results in a faster charge injection. At an extremely high ramp time, the cross-couple devices have enough time to discharge any differential injected charge and the resolved bit value will be a function of only their relative strengths. However, as the ramp rate increases, the time to discharge the injected charge may be insufficient and this residual charge may play a role in the final polarity of the power-up value. This residual charge will be different for different ramp rates and hence the preferred power-up state can differ from one ramp rate to another.

We believe a more complete understanding of the phenomena is still required, including a better theoretical analysis of the power-on behavior of bi-stables and further experimental evaluations. But from our measurements and existing results from literature we can conclude that it is critical that the activation of the supply voltage power-on ramp needs to be tightly regulated to assure maximum reliability during normal operation and to prevent a malicious activation resulting in increased errors.

SA Activation. The reliability of an SA PUF is a function of the offset of the SA elements[8]. Since the ramp rate of the activation signal (sense enable) impacts the effective offset of a SA [62], it is critical to have a tight control on the ramp rate of the sense enable to prevent malicious activation of the SA PUF that could potentially reduce its reliability.

Ramp Control Implementation. There are a number of known methods of generating a controlled voltage ramp on-die including a simple counter and analog-to-digital converter or a more complex direct digital synthesis block. Paired with an on-die voltage regulator and BIST logic, this would enable tight PUF activation control independent of external voltage sources.

4.3 Summary of Chapter

We have presented our results on two *extrinsic* reliability enhancement techniques – multiple evaluation (ME) and activation control (AC). Measured results from the application of these techniques on bi-stable PUFs show a reduction of $\sim$70-80% errors in the PUF response. Hence, these techniques can at best be treated as small-scale reliability enhancement solutions and would still require other techniques to achieve $\sim$100% reliability. However, these techniques are orthogonal to other error correction/reduction techniques and can be implemented with low overheads.

Chapter 5

Intrinsic Reliability Enhancement by Post-Silicon Selection (PSS)

We categorize a reliability enhancement technique as *intrinsic* if it makes a fundamental change to the set of PUF core bits. In this chapter we discuss the first intrinsic technique for reliability enhancement – Post-Silicon Selection (PSS) – which changes the selection criteria of the selected PUF cores from the fabricated pool of PUF cores.

Errors in a PUF response can be reduced by pre-characterizing the PUF cores for reliability and selecting only the ones with the expected reliability higher than a chosen threshold. The PSS flow is shown in Figure 5.1. Before in-field use as a PUF, the PUF cores have their reliability estimated in the enrollment phase. The enrollment phase generates a reliability bitmap which identifies the PUF cores with higher expected reliability. This reliability bitmap is then stored off-chip, typically in a non-volatile memory (NVM). During field evaluation, the reliability bitmap is loaded and responses from only the selected PUF cores are used for the generation of the PUF response.

One way to pre-characterize the PUF cores for reliability in the enrollment stage is to evaluate them multiple times across multiple voltage and temperature combinations and determine which bits are inconsistent and mark them as unusable[15]. However, this may result in unacceptable tester costs and still may not be able to provide any statistical insights about the safety margin in the chosen bits. This method may not be able to predict the PUF reliability at a new untested

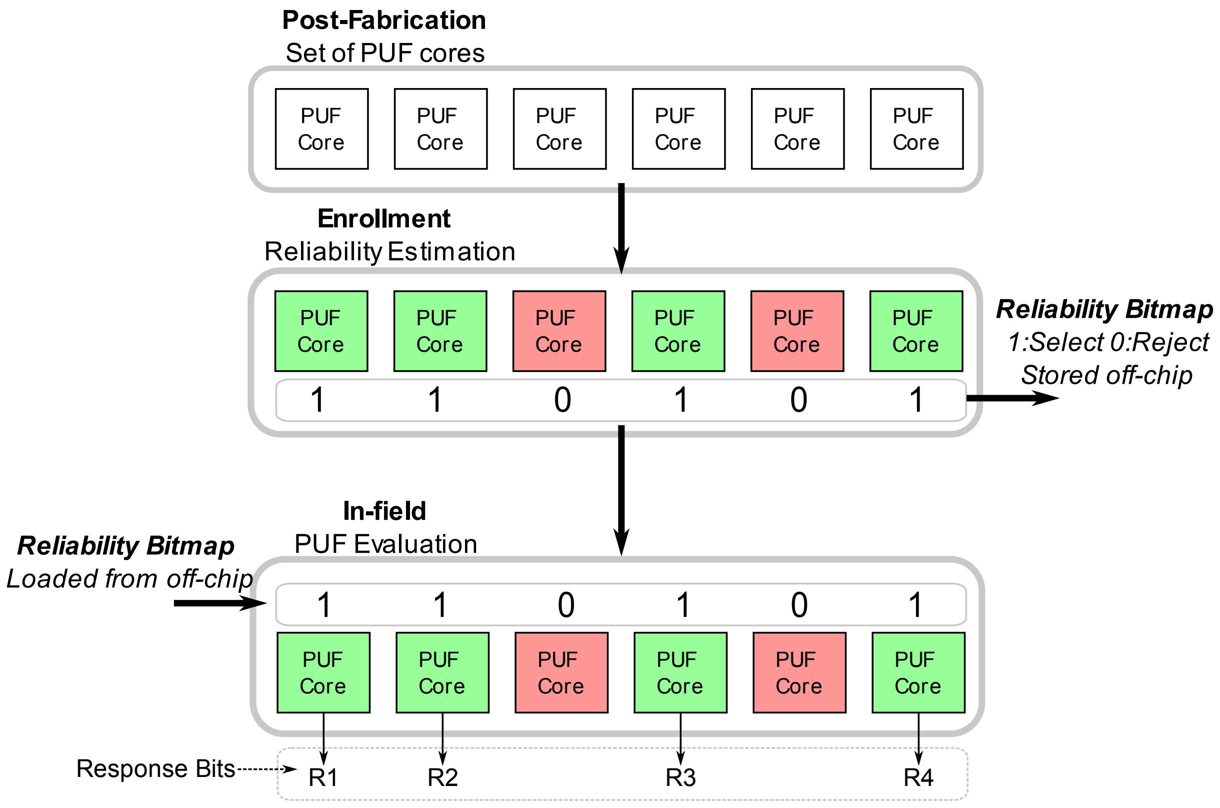


Figure 5.1: A simplified illustration of post-silicon selection (PSS). During the enrollment phase, the reliability of all the fabricated PUF cores is estimated. This reliability information is stored as a reliability bitmap off-chip. The reliability bitmap is loaded during the in-field PUF evaluation and the responses from only the PUF cores with expected high reliability is used.

voltage-temperature corner of operation, under a different noisy environment, or over aging of the PUF. A more efficient way to pre-characterize reliability is to obtain some analog "soft" information from the PUF that has a strong correlation with its reliability. As noted earlier in Section 2, most PUFs amplify some electrical differences to generate their "hard" response bits. Hence, these electrical differences can be good metrics for reliability estimation.

Studies have shown how such "soft" information can be extracted efficiently from the ring oscillator PUFs [65] from their frequency measures. An improved reliability can then be achieved by ensuring that only the ROs with large differences in frequency (typically measured just once at nominal operating conditions) are compared for the generation of the PUF response [46, 65, 71, 72].

We propose the use of sense amplifier PUFs for post-silicon selection [8, 9] and a measure of their offset voltage (V_{OFFSET}) as the "soft" information. As discussed later in this chapter, the V_{OFFSET} of sense amplifiers (i.e., a measure of the inherent bias to resolve to a particular polarity) is a very good metric of their reliability [9]. Use of only the SAs with high magnitude of offset (i.e., the ones with a stronger bias to resolve to one of the polarities) will result in a collection of bits that are more reliable. It must be noted that as shown by our analysis on the comparison of PUF types in Section 3.5, SA-PUFs are superior to RO-PUFs in terms of area, power, and delay per bit of response generation.

In the following sections, we first describe how reliability of a SA PUF can be estimated from its offset voltage measurement. We then describe our implementation of PSS-based key generator followed by experimental results from our Generation II testchip. The key generator is an autonomous and BIST controlled system that implements PSS on SA PUF and generates bits with a bit error rate $<5*10^{-9}$ which is equivalent to a 128-bit key error rate $<10^{-6}$ without using any error correction codes. A 128-bit key failure rate $<10^{-6}$ is a typical targeted failure rate for error correction code (ECC) implementations used in several ECC studies [11, 20, 37, 40, 44]. Finally, as an application of the reliable key generator, we present the framework of a strong-PUF that uses the reliable bits from the key generator in conjunction with an Advanced Encryption Standard (AES), a standard cryptographic primitive.

5.1 PSS in Sense Amplifier PUFs

5.1.1 Sense Amplifier Offset Voltage

As described earlier in Section 2.3.2, sense amplifiers amplify small differential voltages to full swing digital values. In typical memory-based implementations of SAs, correct operation is ensured by providing the SA inputs with a voltage difference larger than the offset voltage ($|V_{OFFSET}|$). Sense amplifiers can be used as PUFs by evaluating them while providing a zero differential input voltage. Under no-noise scenario, and when used as a PUF, it can be expected that a SA with $V_{OFFSET} > 0$ will resolve to a "1" and a SA with $V_{OFFSET} < 0$ will resolve to a "0".

Figure 5.2 shows the histogram of measured offset of 4096 SAs of the latch-style and StrongARM sense amplifiers from our Generation I testchip. The distribution of offset is close to a normal distribution with a mean close to 0mV.

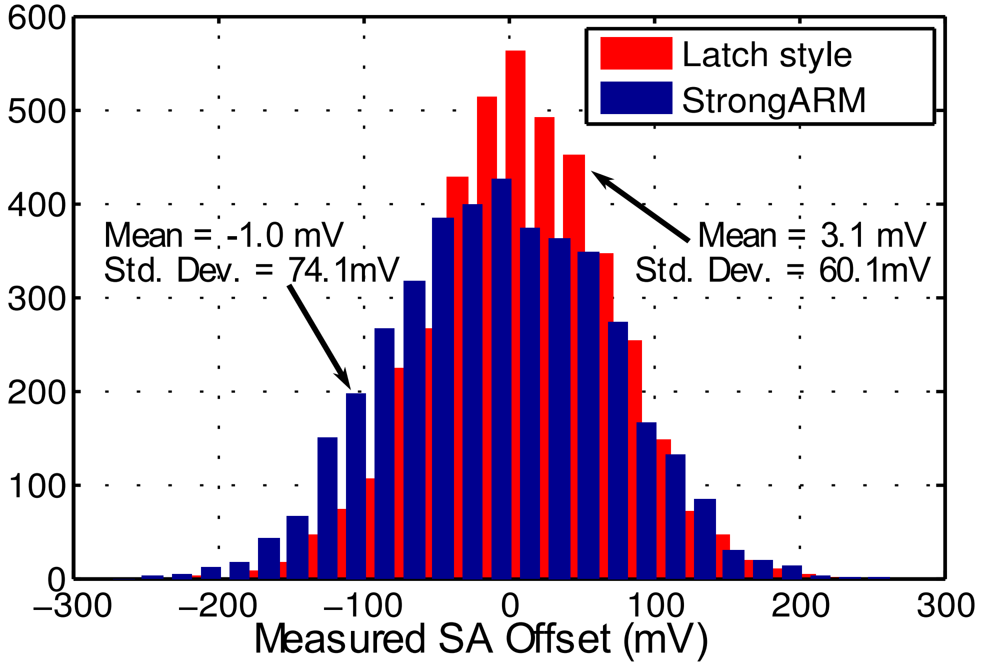


Figure 5.2: Histogram of measured offset from 4096 SAs each of both the latch style and StrongARM sense amplifiers on a chip. The distribution of offset is close to a normal distribution for both the SA types. Measurements from other chips showed similar behavior.

5.1.2 SA Offset Voltage as a Reliability Estimator

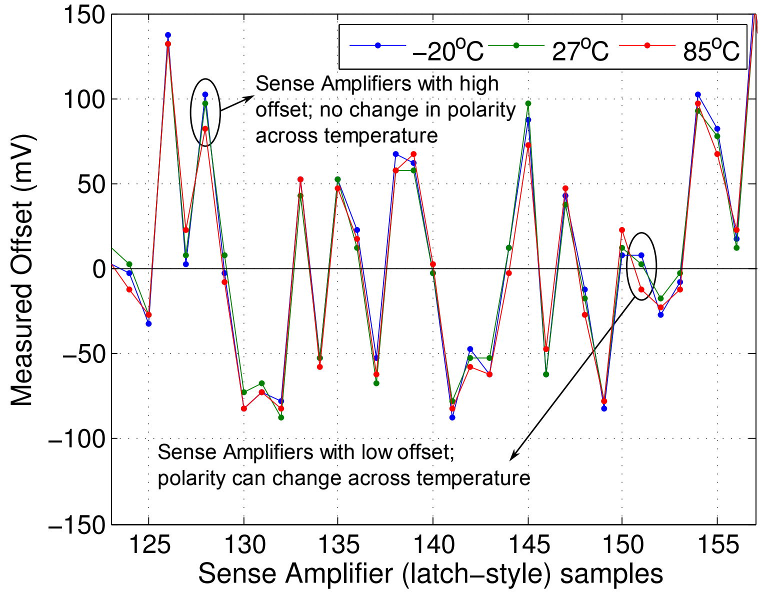
The magnitude of the offset of a SA ($|V_{OFFSET}|$) is a strong indicator of its reliability. This insight follows the observation that SA output depends on the polarity of V$_{OFFSET}$ and the measured V$_{OFFSET}$ changes only slightly across temperature and voltage variations as shown in Figure 5.3. Hence, pre-characterizing SAs for their offset and using only SAs with high |V$_{OFFSET}$| can result in a significant improvement in the reliability of the selected SAs.

Based on this insight, we measured the improvement in reliability in a set of SAs selected based on their $|V_{OFFSET}|$. In Section 5.1.4, we present the measured results from our Generation I testchip. Here, the V_{OFFSET} of each sense amplifier was measured using a procedure as detailed in Appendix B. The estimation of reliability enhancement is done by post-processing of the results in software. Promising results from the Generation I testchip encouraged us to build an autonomous, BIST controlled key generator system in our Generation II testchip that generates the reliability bitmap in hardware (during enrollment) and autonomously aggregates reliable bits using the reliability bitmap (during runtime). This key generator system is detailed in Section 5.2 and large-scale experimental results from the system are presented in Section 5.2.2.

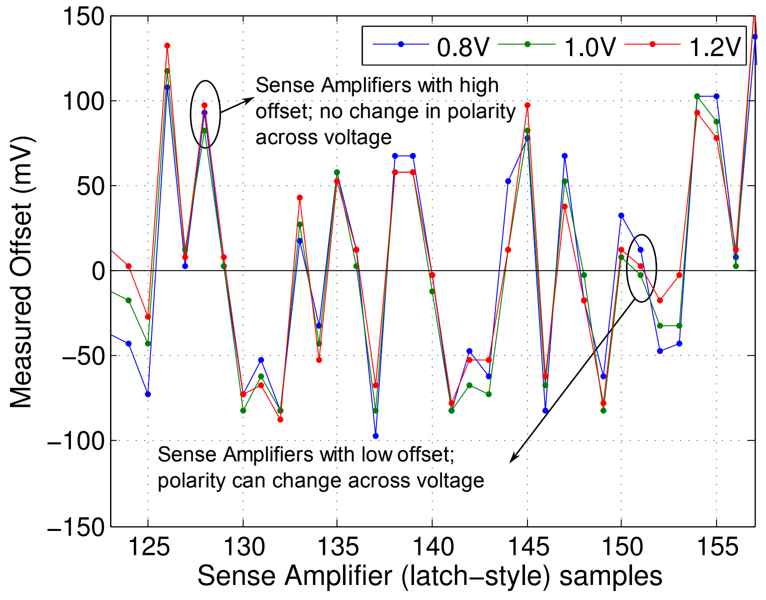
5.1.3 Efficient Reliability Characterization

One way of characterizing reliability of SA PUFs is to measure the V_{OFFSET} of each SA (as detailed in Appendix B) and then selecting the SAs with $|V_{OFFSET}| > \Delta V_{IN}$ where ΔV_{IN} is a threshold that provides sufficient reliability across different and noisy environmental conditions. This method would require a large number of cycles depending on the resolution of offset measurement. A more efficient method to characterize reliability is as follows.

Figure 5.4(b) shows the two-phase enrollment operation used to characterize the reliability of a SA PUF. If a large number of SAs are arrayed with their inputs shorted across all of them (i.e., a common $V+$ and a common $V-$), then the entire array of SAs can be characterized together. In the first phase (*POS*), the inputs are configured such that $\Delta V_{IN} = (V+) - (V-)$ and in the second phase (*NEG*), the connections are reversed so $\Delta V_{IN} = (V-) - (V+)$. As shown in Figure 5.4(b), at the end of enrollment, a SA is selected as a potentially reliable one if the output



(a) V_{OFFSET} at different temperatures



(b) V_{OFFSET} at different voltages

Figure 5.3: Measured V_{OFFSET} for a sample of latch-style sense amplifiers at different (a) temperatures and (b) voltages. V_{OFFSET} does not show a significant variation across voltage or temperature variations and hence, when V_{OFFSET} measured at nominal conditions (27°C and 1.0V for GP devices) has a large magnitude, it is more likely that the V_{OFFSET} and hence the SA PUF response bits will have the same polarity across all voltage and temperature variations.

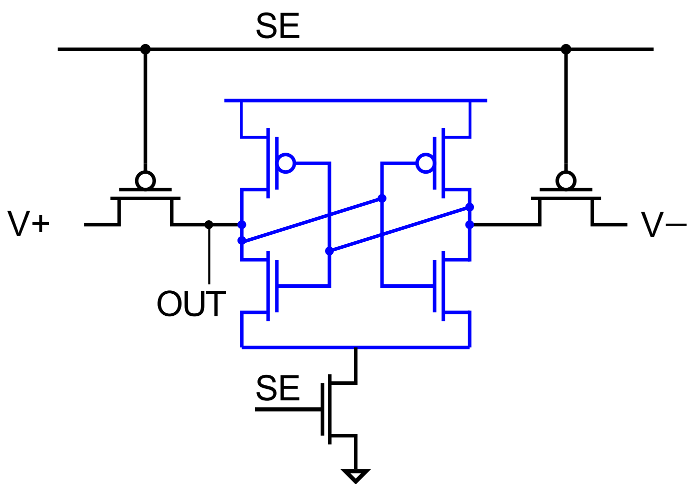
of the SA is consistent (either 1 or 0) for both the phases. A consistent output of a SA is an indicator that its $|V_{OFFSET}| > \Delta V_{IN}$ (i.e., an external ΔV_{IN} was insufficient to make a SA flip its preferred polarity) and hence the SA has a high probability to resolve to a consistent polarity when $\Delta V_{IN} = 0$ across different and noisy environmental conditions. $V+$ and $V-$ are kept fixed at a voltage differential (ΔV_{IN}) that provides sufficient robustness over environmental variations.

Our experiments from our Generation I testchip show that a ΔV_{IN} ~50mV results in selection of ~50% of SAs which have extremely high reliability [9]. These results are shown next in Section 5.1.4. Note that this threshold of ΔV_{IN} can vary for different SA designs (topology, sizing, or layout), for different process technologies, and for different measurement conditions.

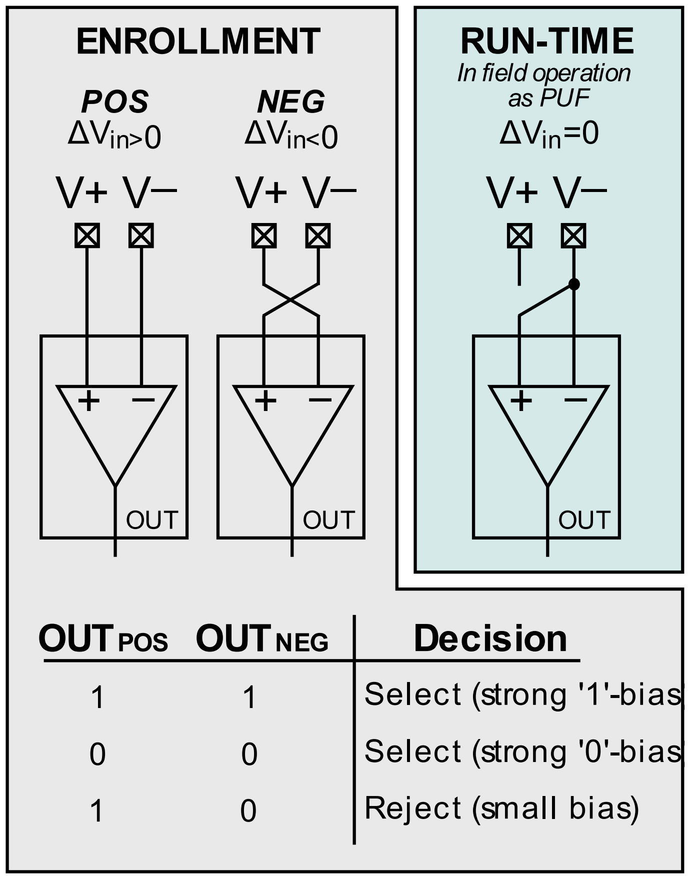
5.1.4 Reliability Enhancement (Results from Generation I Testchip)

Figure 5.5 shows the reduction in errors in selected SAs with a more restrictive thresholds of selection. The increased threshold will, however, result in the rejection of a higher percentage of all the SAs fabricated, and hence a lower percentage utilization of SAs on chip. Higher the threshold of selection, higher will be the overall reliability of the selected SAs, but at the cost of a reduced percentage utilization of SAs. This tradeoff is shown in Figure 5.6. For example, with a $|\mathrm{V}_{OFFSET}|$ threshold of 20mV, the percentage of errors reduced from 18.8% to 4.1% for latch-style SAs after rejecting 33% of SAs (67% SA utilization). The reliability-to-V_{OFFSET} correlation enables a PUF designer to have an efficient and deterministic method of selecting reliable SA elements based on the expected noise and environmental variations.

Note that the results presented in Figure 5.5 are using one evaluation at every voltage/temperature corner. Applications that require high reliability, e.g., key generation, require the final key to have a key error rate $< 10^{-6}$ or an equivalent bit error rate $< 7.8 * 10^{-9}$. To demonstrate such low error rates in a PUF that uses PSS, large-scale measurements would be required. We present such large-scale silicon measurements in Section 5.2.2 to demonstrate that using PSS in SA PUFs can achieve the desired key error rates without using any ECC.



(a) Latch-style sense amplifier



OUT POS	OUT NEG	Decision
1	1	Select (strong '1'-bias
0	0	Select (strong '0'-bias
1	0	Reject (small bias)

(b) Enrollment and run-time operations

Figure 5.4: (a) Latch-style sense amplifier schematic, with bistable portion highlighted in blue. (b) Enrollment (pre-characterization) and in-field operation (as PUF).

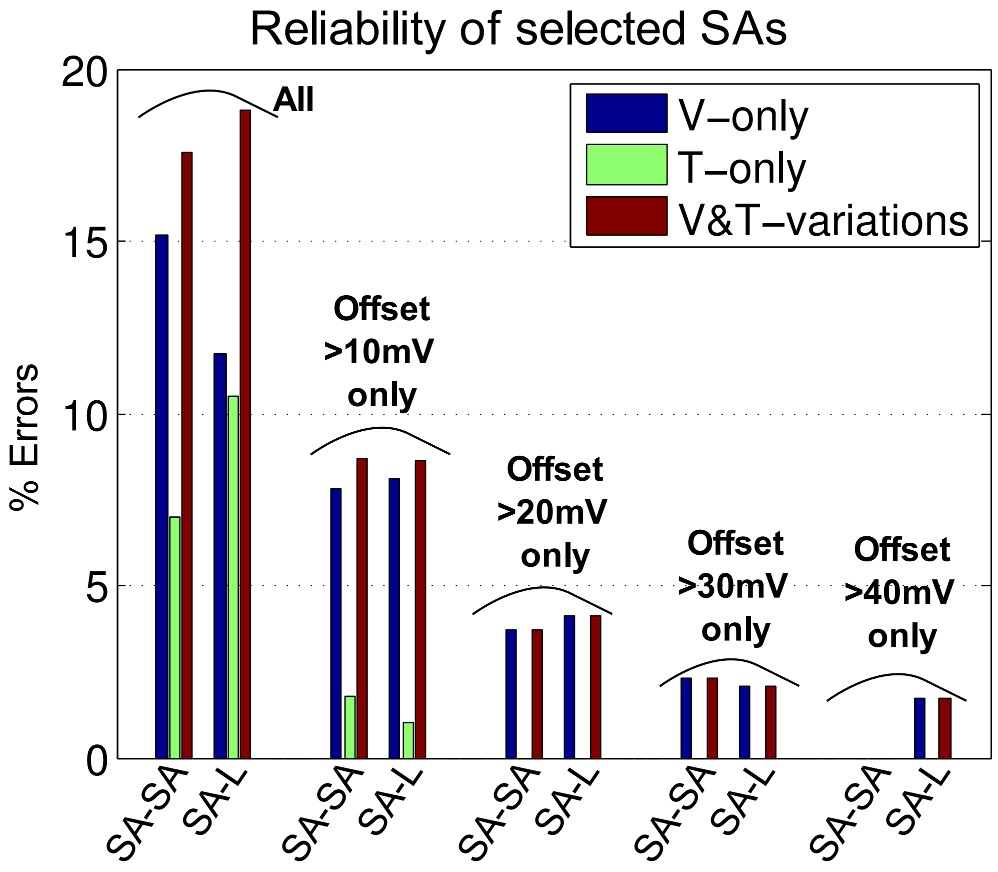


Figure 5.5: Reliability of SA PUF (expressed as errors in generated response bits) using PSS. Results presented are from test-structures built in Generation I testchip. Percentage errors reduce when SAs with higher $|V_{OFFSET}|$ are chosen. Note that the percentage errors for the case where all SAs are selected is consistent with the error percentages as shown in Figure 3.15.

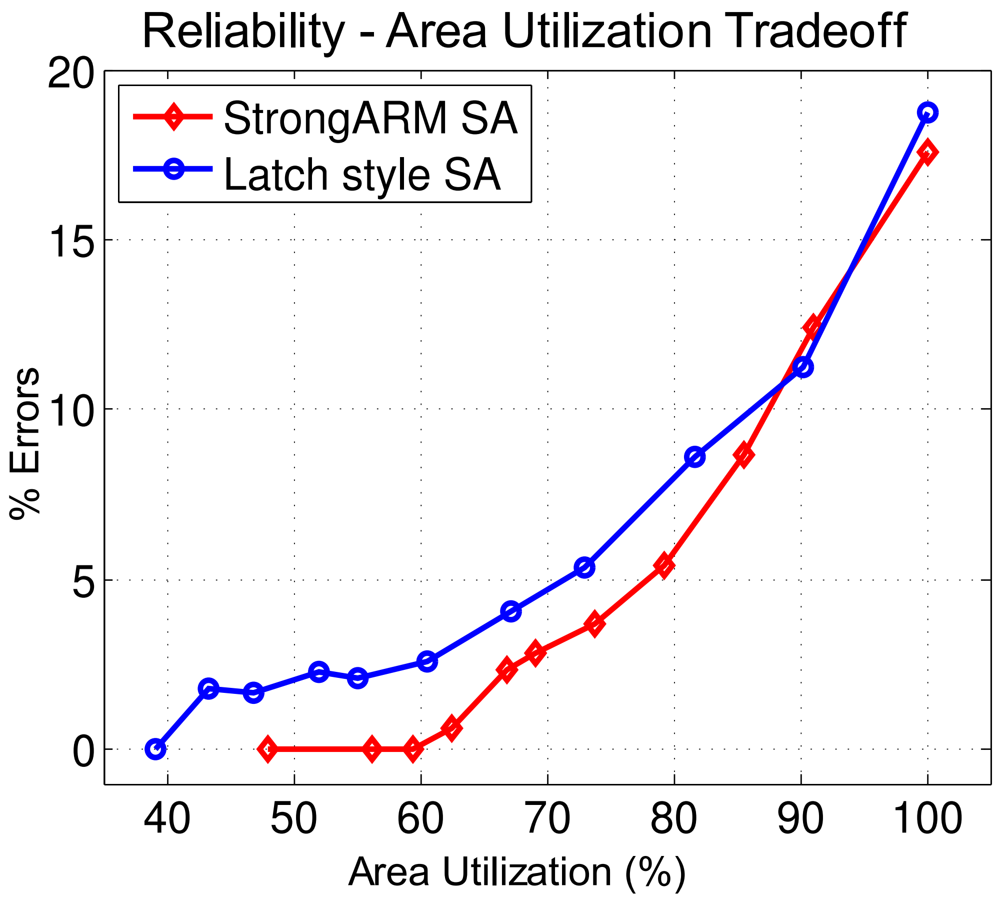


Figure 5.6: Percentage utilization-reliability tradeoff with PSS in SA PUFs. With a more restrictive selection of SAs (i.e., choosing only the ones with a higher $|V_{OFFSET}|$) the percentage errors in the selected bits is reduced (as shown in Figure 5.5), but at the cost of a lower percentage utilization of the fabricated SAs. No errors were found in $\sim$40% of the fabricated latch-style SAs with highest V_{OFFSET} are selected or $\sim$60% of the fabricated StrongARM SAs with highest V_{OFFSET} are selected.

5.2 Autonomous Key Generator and Secure Strong-PUF

5.2.1 Design

Figure 5.7 shows the top-level schematic of our self-contained BIST-controlled key generator. It consists of a 64x64 SA PUF array and two 64x64 SRAM arrays. The layout snapshot of the implementation of the key generator and the AES is shown in Figure 5.8. The SA PUF array is arranged and designed much like a typical SRAM array and each row of the SA array is activated by a rising sense enable (SE) signal (Figure 5.4(a)) which is implemented like the word-line (WL) signal in a SRAM array. Figure 5.9 describes the key generator operation. During the first phase (*POS*) of enrollment (pre-characterization), the values from the SA PUF array (OUT_{POS} in Figure 5.4(b)) are read and temporarily stored in the SRAM array named 'Value Array'. In the second phase (*NEG*), the values from the SA PUF array (OUT_{NEG} in Figure 5.4(b)) are compared with OUT_{POS}) by simultaneously accessing the 'Value Array'. If $OUT_{POS} = OUT_{NEG}$ for a SA, suggesting that $|V_{OFFSET}| > \Delta V_{IN}$, its location is marked as potentially reliable by storing a '1' at the corresponding location in the Reliability Bitmap SRAM array. The written word into the Reliability Bitmap is generated by a bit-wise XNOR of OUT_{POS} and OUT_{NEG}. The end product of enrollment is the completely filled Reliability Bitmap array which is equivalent to the helper data of typical ECC schemes. This reliability bitmap needs to be stored in a non-volatile memory with the PUF or needs to be sent to the PUF from a server before it needs to generate its key in the field.

Figure 5.9 shows the execution steps at run-time when the key generator is used in the field. First, the reliability information is loaded into the Reliability Bitmap array. Then the SA PUF array is activated with $\Delta V_{IN} = 0$ (Figure 5.4(b)) while the corresponding reliability information is also read from the Reliability Bitmap array. The Reliable Bit Aggregator aggregates the SA values from the first *N* reliable locations (as per the Reliability Bitmap) to generate and stores a reliable *N*-bit key in registers (Figure 5.10). In our design, we chose *N*=512 and the first 128 bits are used as a key to an AES.

Note that the helper data (Reliability Bitmap block) carries no information about polarity of the bits but only the physical location of the potentially more reliable bits. Hence they do not leak

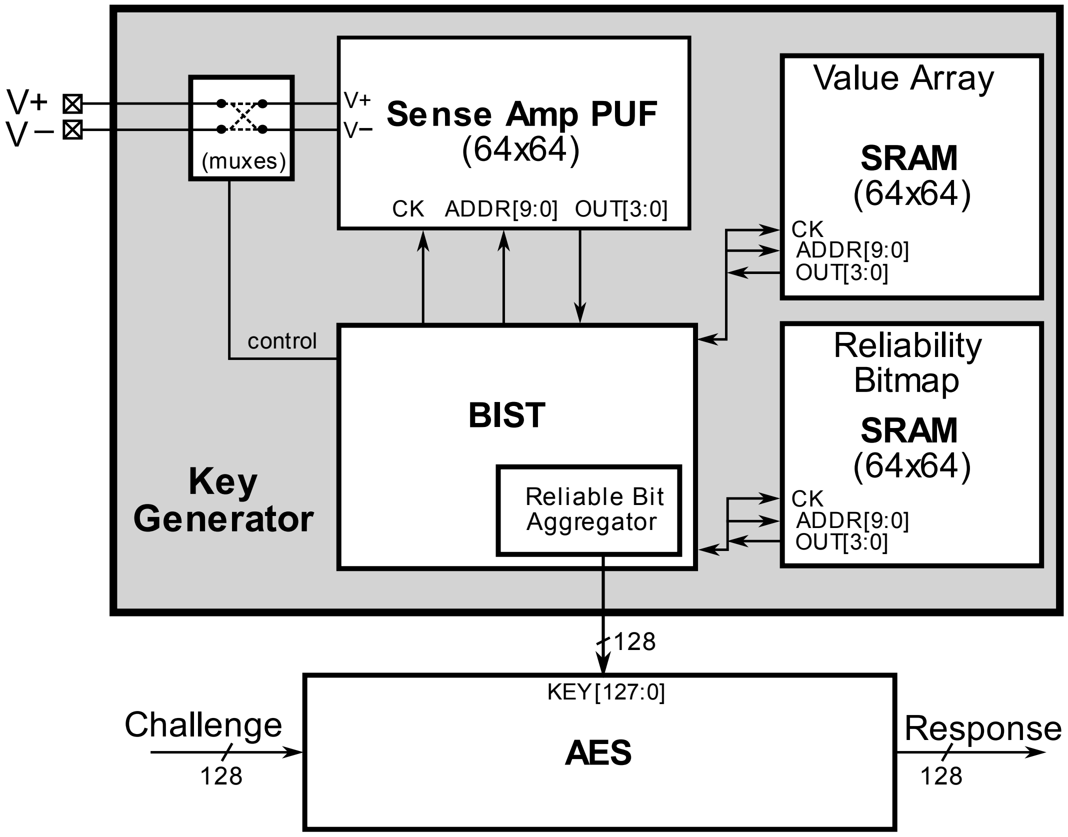


Figure 5.7: Key generator and strong-PUF design. A self-contained BIST controlled sense amplifier (SA) PUF based key generator. The BIST automatically generates the reliability bitmap in the enrollment phase which is then used at run-time to select reliable bits from the SA PUF array. The first 128 of these bits are used as the key in an AES primitive to realize a reliable and secure strong-PUF. The input and output of the AES primitive are treated as the challenge and response respectively.

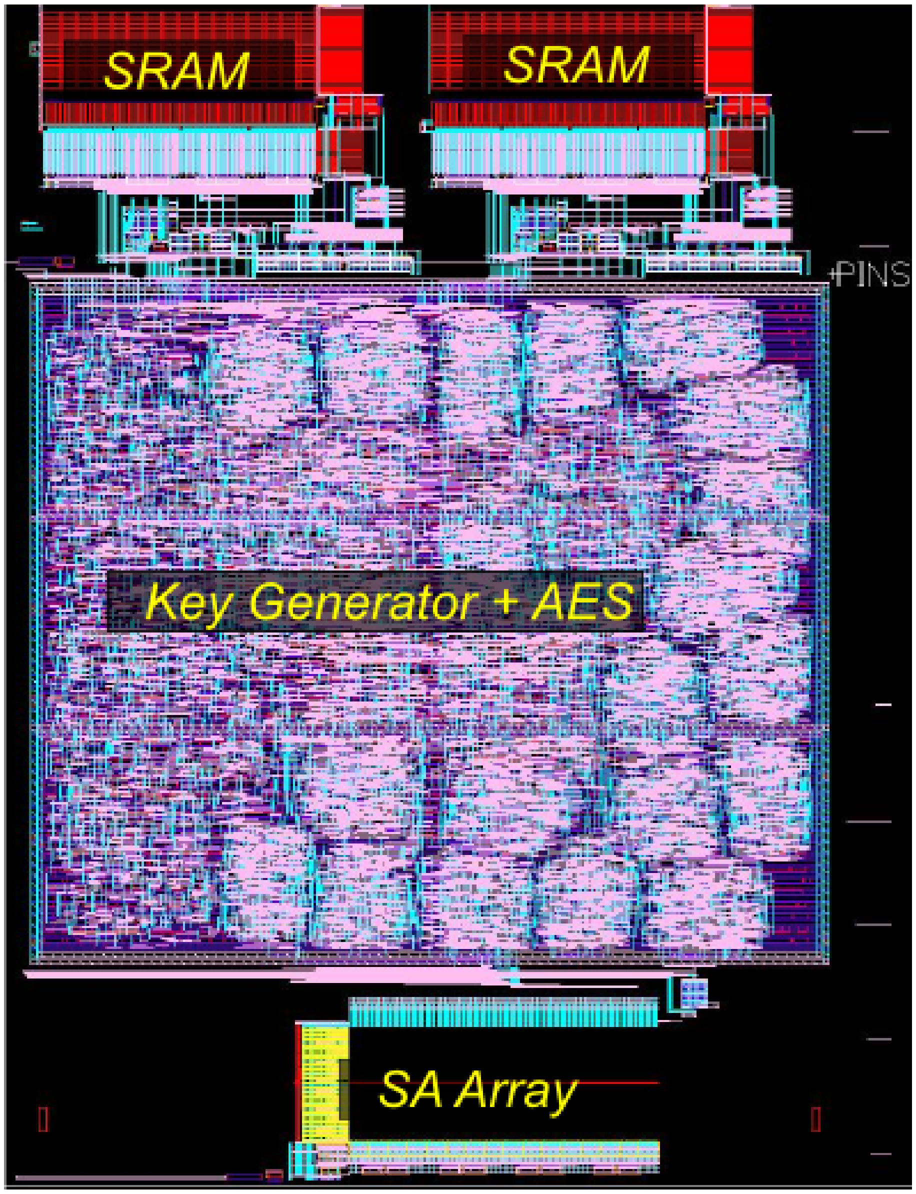


Figure 5.8: Layout snapshot of key generator and AES for strong PUF realization.

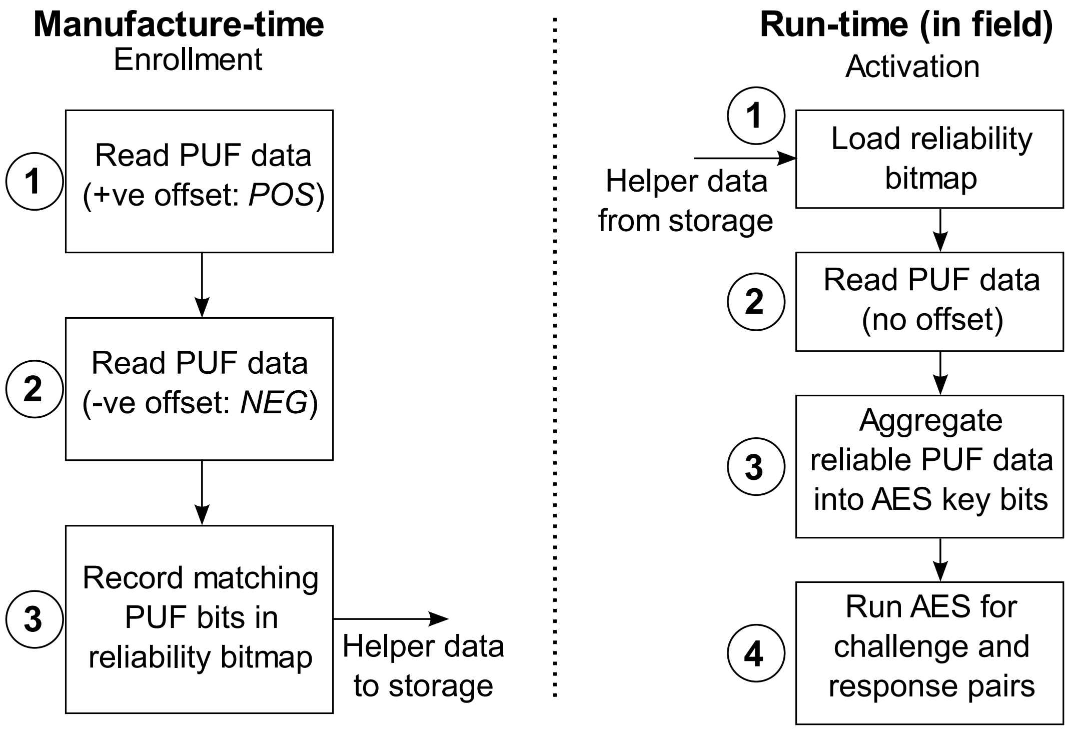

Manufacture-time
Enrollment
1 Read PUF data (+ve offset: *POS*)
2 Read PUF data (-ve offset: *NEG*)
3 Record matching PUF bits in reliability bitmap
Helper data to storage
Run-time (in field)
Activation
Helper data from storage
1 Load reliability bitmap
2 Read PUF data (no offset)
3 Aggregate reliable PUF data into AES key bits
4 Run AES for challenge and response pairs

Figure 5.9: Flowchart of key generation and strong-PUF realization.

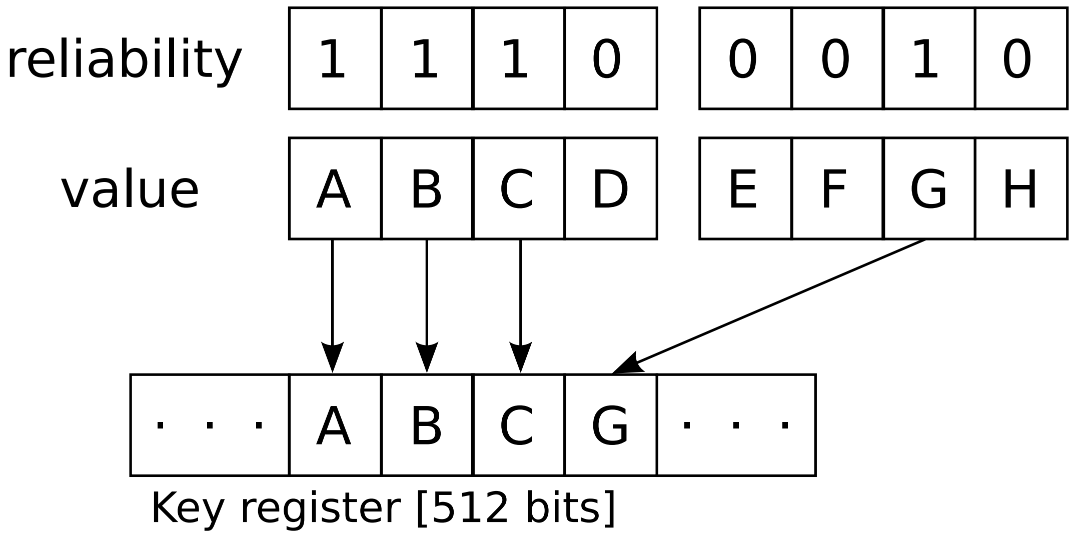


Figure 5.10: Demonstration of the repeated aggregation of reliable PUF bits into the AES key. Letters used instead of bits in value for clarity. 512 bits stored in key register for analysis purposes; the first 128 of these are used as the AES key.

any information about the bits unless there was any correlation found in the bits generated from the SA PUFs in the array. However, the biggest contributor of V_{OFFSET} (and hence the polarity of bit) is *local* random variations in the devices of a SA (e.g., random dopant fluctuations and line edge roughness) and hence the bits of the array can be assumed to be largely independent.

5.2.2 Experimental Results (Generation II testchip)

In this section we provide measured results of the key generator from our Generation II 65nm CMOS testchip (Appendix A.2). First, we present the reliability measures followed by the area and speed measurements of the key generator as well as for our strong-PUF realization.

5.2.3 Reliability of Generated Key

Depending on the choice of ΔV_{IN}, the key generator creates a Reliability Bitmap that is used to select a set of bits for use in-field. A higher ΔV_{IN} will result in a smaller set of selected SAs, but one with higher expected reliability. Hence, reliability for our design will be a function of ΔV_{IN}. Figure 5.11(a),(b) show the measured outputs from the SA PUF array after the *POS* and the *NEG* phases in enrollment for different values of ΔV_{IN}. Figure 5.11(c) shows the reliability bitmap that is generated after the two phases. Each square in Figure 5.11 consists of 64x64 pixels that represent the contents of a 64x64 array. A '1' in the SA PUF output is represented by a white pixel and a '0' in the PUF output is represented by a black pixel. We repeated the *POS* and *NEG* phases 100 times, and some of the PUF outputs are noisy, i.e., they sometimes resolve to a '0' and sometimes to a '1'. For illustration, such SA PUFs are represented by a grey pixel and the ratio of '1's and '0's is indicated by the greyscale value. However, the reliability bitmap is generated using only the first evaluation of the SA PUF. A white pixel in the Reliability Bitmap represents a bit location to be selected (as potentially reliable) and a black pixels represent the locations of the unselected bits. We see that for a higher ΔV_{IN}, the SA PUFs tend to generate more 1's in the *POS* phase (more white pixels in Figure 5.11(a)) and more 0's in the *NEG* phase (more black pixels in Figure 5.11(b)). Also, for higher ΔV_{IN}, there is a more rigorous selection criterion for reliability, as is seen by the less number of white pixels in the Reliability Bitmap. This is further illustrated in Figure 5.12 which shows the percentage of SA locations that are selected during enrollment for different values of ΔV_{IN}. For example, for ΔV_{IN}=50mV, 38.4% of 4096 bits (=1573) bits were selected in the reliability bitmap and for ΔV_{IN}=120mV, 3.7% of 4096 bits (=151) bits were selected in the reliability bitmap.

For reliability, we first take 10,000 measurements (run-time operation as shown in Figure 5.4(b) and Figure 5.9) at all combinations of voltage: 1.0V, 1.2V, 1.4V and temperature: -20°C, 27°C, 85°C. We then take 180,000 measurements at the corner with worst reliability as measured from the initial 10,000 runs (found to be 1.0V, 85°C). We also take 100,000 measurements at the nominal corner (1.2V, 27°C). The bit error rate (BER) and the key error rate (KER) is computed as follows. Any selected SA that resolves inconsistently in any of the measurements is considered an unreliable SA and every inconsistent SA increases the bit error count by 1. The BER is then ratio of bit error count, averaged across all runs, to the total number of selected bits. For a key to be error free, all the bits of the key must be error free. We assume the secrecy rate to be 0.75 (i.e., entropy per bit = 0.75), using the results from previous studies on bi-stable based PUFs [18, 37]. Hence, to obtain a 128-bit key, $\lceil 128/0.75 \rceil = 171$ bits need to be generated which can be later compressed (privacy amplification) to 128-bits of perfect entropy. Hence for

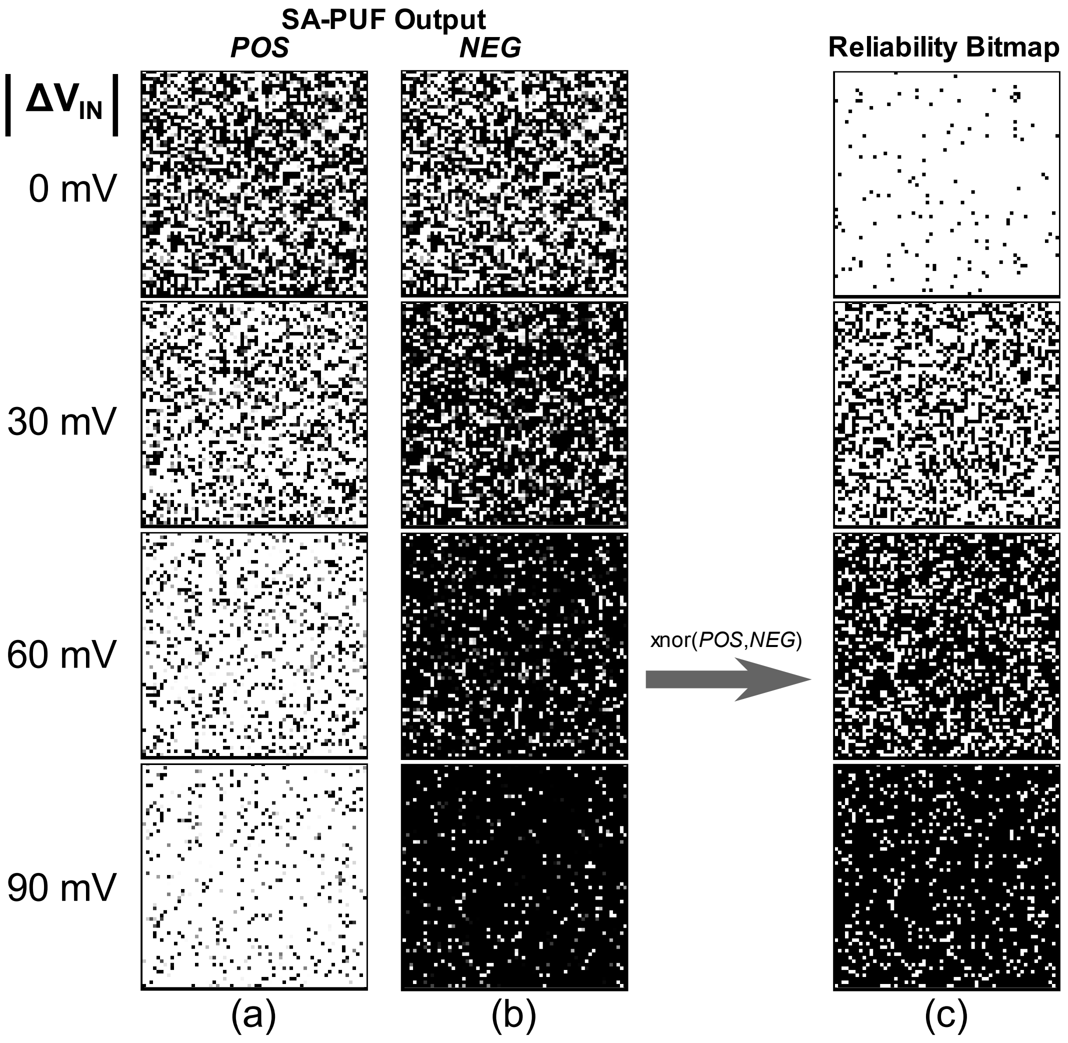


Figure 5.11: SA PUF response map in the (a) *POS* and the (b) *NEG* phases.The percentage of '1's and '0's is $\sim$ 50% for ΔV_{IN}=0mV. With an increasing ΔV_{IN}, the response maps in the two phases show a bias towards a higher percentage of '1's and '0's in the *POS* and the *NEG* phase respectively. (c) The reliability bitmap shows a reduced number of selected SAs for higher ΔV_{IN}.

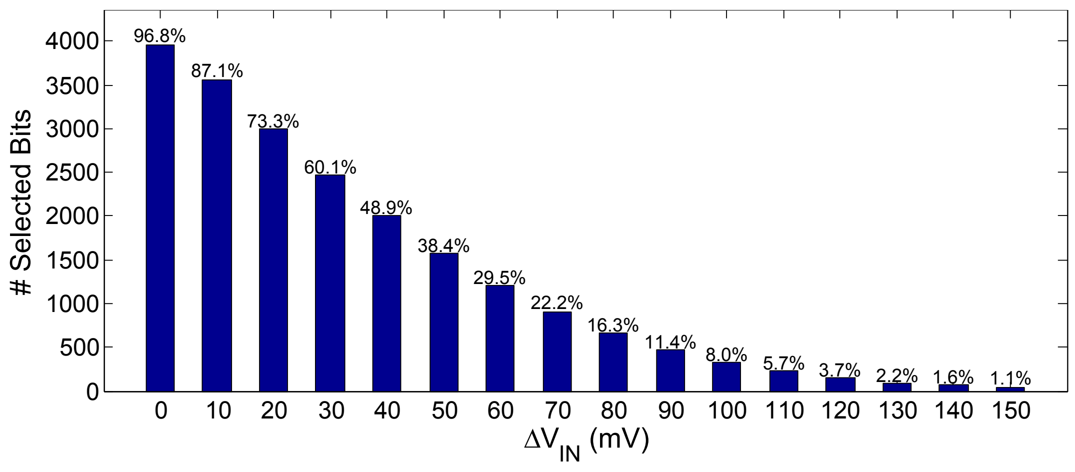


Figure 5.12: Percentage of bits (of the 4096 bits in the SA PUF array) that are selected in the enrollment stage i.e. percentage of 1's in the Reliability Bitmap for various ΔV_{IN}

this study, we define the key error rate $KER = 1 - (1 - BER)^{171}$.

Figure 5.13 shows the measured errors in the selected set of SAs for different ΔV_{IN}. As expected, a higher ΔV_{IN} results in reduced number of errors. Figure 5.13(a) shows the errors across temperature variations at constant voltages of operation. Figure 5.13(b) shows the errors across voltage variations when operated at constant temperatures. Figure 5.13(c) shows the errors across all voltage and temperature variations as measured from the first batch of 10,000 measurements across all corners. We find that when enrollment is done using $\Delta V_{IN} = 60mV$, no errors were found in any of the 1213 selected bits in 90,000 measurements.

Large-scale silicon measurements. The measurements from the additional 180,000 and 100,000 measurements at the worst case corner (1.0V, 85°C) and the nominal corner respectively, *found no errors in any of the runs*. We define worst case bit error rate (BER_{WC}) as follows. If no errors were found in N runs of S selected bits, then we conservatively assume that the first error would occur in the very next measurement and define our $BER_{WC} = 1/(N * S + 1)$. Due to the pessimistic definition we can safely claim that for our design the measured BER is at least lower than BER_{WC}. Since no error were observed in any of the 1213 SAs in the 180,000 measurements at worst case, our $BER_{WC} = 4.6 * 10^{-9}$ and we can claim that experimentally observed $BER < 4.6 * 10^{-9}$. This is equivalent to a $KER < 0.8 * 10^{-6}$.

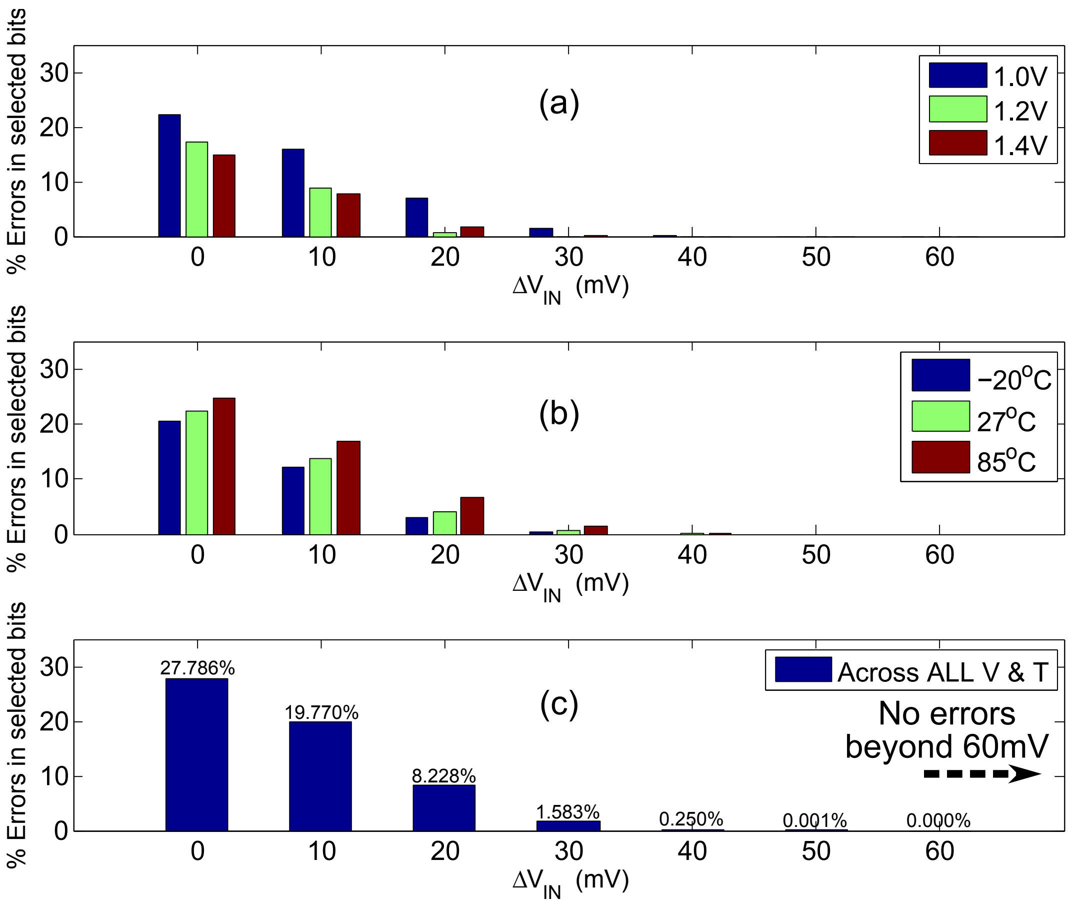


Figure 5.13: Percentage Error bits in the selected set of SAs. For the error measurements, the SA PUF array was evaluated 10,000 times at all combination of voltage (1.0V, 1.2V, 1.4V) and temperature (-20°C, 27°C, and 85°C) (nominal: voltage=1.2V; temperature = 27°C), hence a total of 90,000 evaluations. (a) Errors across temperature variations while keeping the voltage constant. (b) Errors across voltage variations while keeping the temperature constant. (c) Errors across all voltage and temperature variations.

5.2.4 Error Modeling

We model-fit the measured error rates to the various values of ΔV_{IN}. For all the 9 environment corners, we found that the rate of reduction of BER was super-exponential w.r.t. an increasing ΔV_{IN}. We were able to achieve a good fit by choosing the following model: $log(y) = ae^{bx}$ where $y = BER$ and $x = \Delta V_{IN}$. The model fits very well for all the observed errors at all the 9 corners. The measured BER, the model, and the worst case bit error rate (BER_{WC}) are shown in Figure 5.14. The model predicts that the BER, for the worst case corner and with a ΔV_{IN}=65mV is 7*10^{-12} or an equivalent KER $\sim$1.2*10^{-9}. Clearly, there must exist an error floor due to various error phenomena including soft particle strikes that would result in errors. However, at such error levels, the PUF would then be just as unreliable as any other part of the digital circuitry.

5.2.5 Uniqueness

Uniqueness is a measure of how uncorrelated the response bits are across chips, and ideally the response bits should differ with a probability of 0.5. The Hamming distance (HD) of a k-bit response from ideally unique chips should follow a binomial distribution with parameters $\mathcal{N} = k$ and $p = 0.5$ and the mean of the HD distribution should be equal to k/2. For uniqueness measurements, we create 256 16-bit words from the 4096 SA raw bits from 15 different chips for HD computation. Figure 5.15 shows that the pair-wise HD of response bits from three arbitrarily chosen chips. The HD distribution is close to ideal and the mean of HD (ideally 8.00 for k=16) for all pair-wise combinations taken from 15 measured chips (i.e., total 105 combinations) was found to be in the range 7.69–8.26.

5.2.6 Randomness of the Selected Bits

We compute the percentage of 1's in the selected set of bits for various ΔV_{IN} just to ensure that the selection process does not favor bits with a certain polarity. We find that the percentage of 1's are within the range 43%–50% (Figure 5.16).

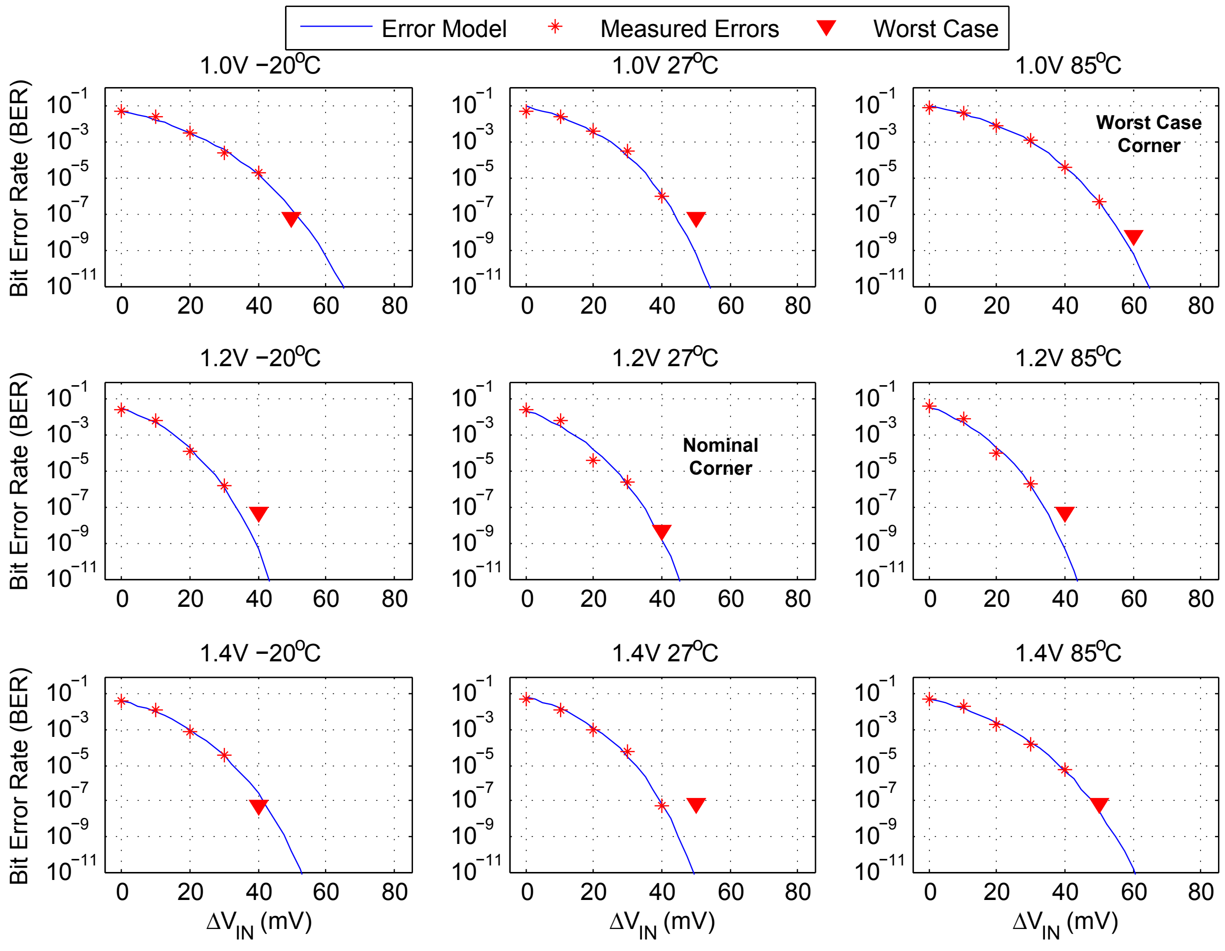


Figure 5.14: Measured bit error rate (BER) and modeling of errors beyond the measured range. The measured errors are from 180,000 measurements in the worst case corner, 100,000 measurements in the nominal corner, and 10,000 measurements at all other corners. If no errors are observed in *N* measurements of *S* selected bits, then the worst case bit error rate is pessimistically defined as 1 error in *N*S* measurements. The following error model was a very good fit: $log(y) = ae^{bx}$ where $y = BER$ and $x = \Delta V_{IN}$.

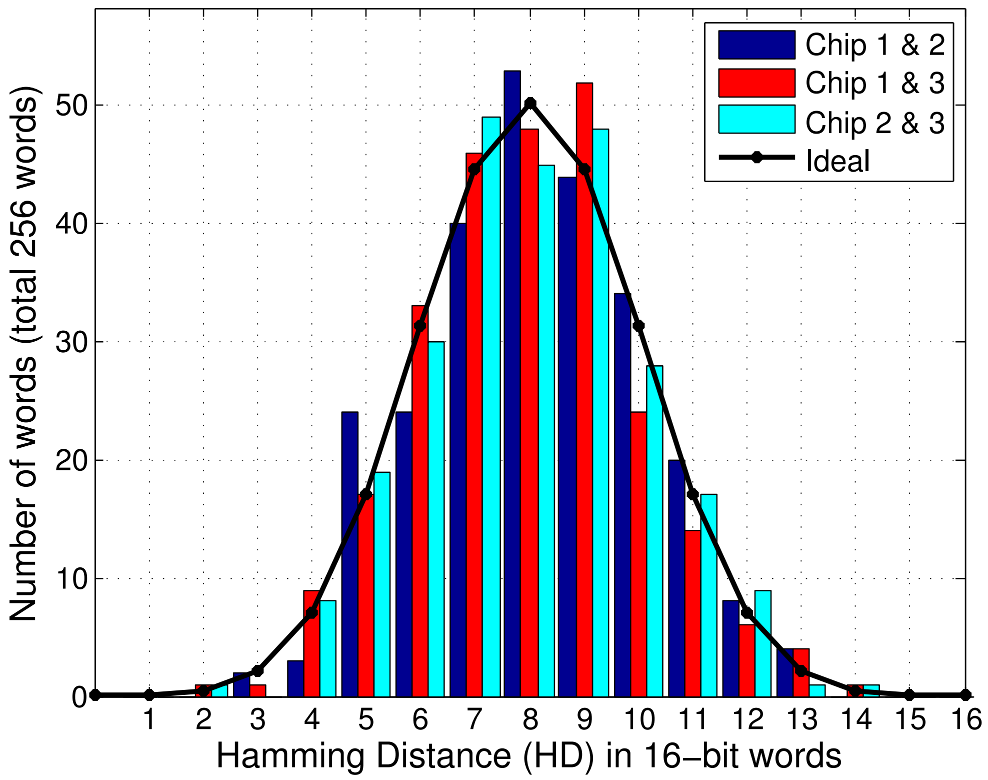


Figure 5.15: Histogram of Hamming distance (HD) of response words from the SAs across three chips. Also shown is the probability mass function of the HD in responses from ideally unique chips. For the HD comparison, 256 16-bit words are created from 4096 bits of the SA arrays. The pair-wise HD of response bits from the three chips is close to ideal with means of 8.16 7.93, and 7.99.

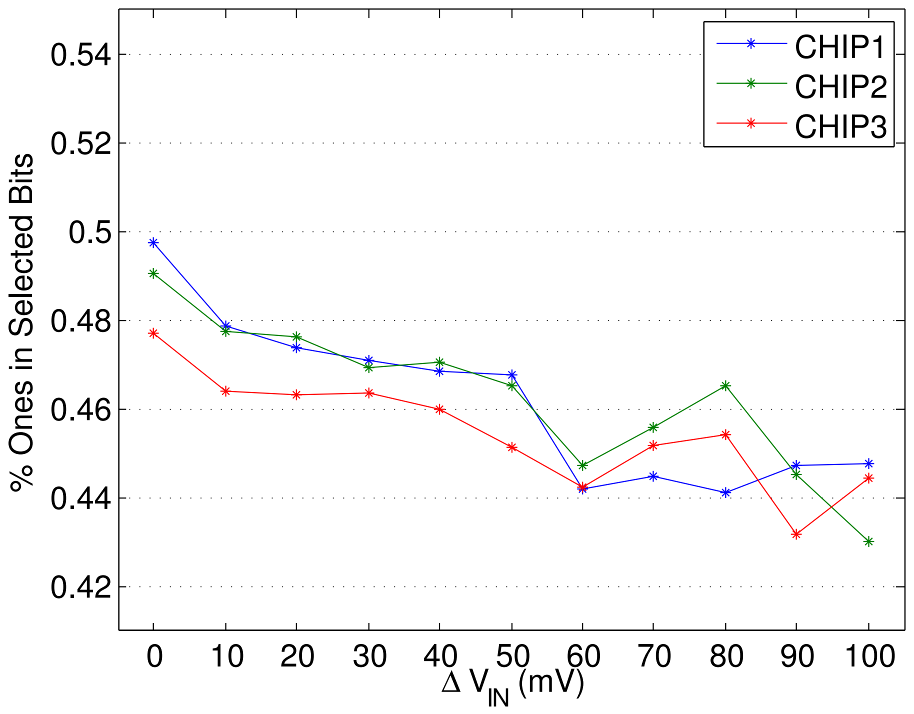


Figure 5.16: Percentage of 1's in the selected bits for various ΔV_{IN} for 3 different chips.

5.2.7 Speed

Speed of Enrollment of SA PUFs. The enrollment needs to be done just once to extract the reliability bitmap of the SA PUF array. For this design, the enrollment is a completely self-contained operation and requires no configuration except fixing the two signal pins $V+$ and $V-$ to provide a sufficient ΔV_{IN} which could be done with an internal resistive ladder or other bias generator circuits. Enrollment takes 1024 cycles each for the *POS* and the *NEG* phase with a pause of *K* cycles between to allow the inputs of the SAs to settle to the switched voltage. From simulations of the design with parasitic capacitance extracted, we estimated that the SA inputs take $\sim$250ns to settle. We ran our design at 10 MHz and *K* was set to 16 pessimistically to allow 1600ns for the inputs to settle. Although we tested our design in silicon at only 10 MHz due to test equipment limitations, in simulations the key generator is able to run at up to 250 MHz. At the tested frequency of 10 MHz, the enrollment is achieved in 206.4 μs. At the simulation frequency of 250 MHz, and still allowing 1600ns for the inputs the settle, the enrollment delay is 9.8 μs. Further, if the word-size of the SA PUF and the SRAM arrays is increased from 4, to say 32, the number of cycles would reduce by a factor of 32/4 = 8. However, increasing the word-size would require more I/O pins since this reliability information has to be sent off chip for storage.

Speed of Key Generation. At run-time, the key generator first loads the Reliability Bitmap Block (generated during enrollment and stored off-chip). Then the SA PUF is evaluated while the Reliable Bit Aggregator accumulates the reliable bits of the key by processing the output words from the SA PUF array and the Reliability Bitmap array, one word at a time (Figure 5.10). The run-time operation for the key generation takes 2048 cycles in the worst case: 1024 cycles to load the Reliability Bitmap Block, and 1024 cycles to aggregate 171 reliable bits from the SA PUF assuming there are exactly 171 reliable bits in the PUF. Note that we are assuming a secrecy rate of 0.75 and hence a total of 171 bits required to generate high entropy 128 bits. If the required 171 reliable bits can be extracted from the first *N* bits of the SA PUF, then this information can also be stored during enrollment to limit the bits loaded and then processed for the generation of the required number of reliable bits. For the worst case (loading and processing all 4096 bits), it takes 204.8μs at the measured frequency of 10 MHz and 8.2μs at the simulated frequency of 250 MHz. From data gathered from the our test chips, we see that $\sim$1200 of the 4096 bits (i.e., $\sim$30%) bits show extremely high reliability for $\Delta V_{IN} = 60mV$ (as shown in Figure 5.12 and Figure 5.13). Thus the estimated number of bits to be read to generate 171 reliable bits is 570 and that would require 143 cycles each for loading and processing the required number of bits. Hence, in this case, the generation of the key would take 28.6μs at the test frequency of 10 MHz and 1.14μs at the simulated frequency of 250 MHz. These delay numbers compare favorably to a recently published FPGA implementation which has a PUF key-generation time of 5 ms (running at 54 MHz) [42].

Speed of the strong-PUF (response time). The AES primitive implemented in our system requires 10 cycles to generate a 128-bit response for a 128-bit challenge. At the simulated frequency of 250 MHz, this is 40 ns per challenge/response pair (CRP).

Design Block	GE	Area on Chip
Synthesized logic	14,452	93,700 μm^2†
→AES	10,487	34,436 μm^2
→IO-Ctrl	1,493	10,844 μm^2
→Test-Ctrl	2,470	13,198 μm^2
Value/Reliability SRAMs	–	2 x 7,700 μm^2
Sense Amp PUF	–	9,700 μm^2
Total area	–	118,800 μm^2

Table 5.1: Area of the key generator. † The total area of the synthesized logic is more than the sum of the area of different synthesized blocks as it includes the place and route overhead as well as several scan-flops for test purposes.

5.2.8 Area

Table 5.1 shows the area consumption of the key generator and the realized strong-PUF and the gate equivalents (GE) for the synthesized blocks. The total on-die area of the design is 118.8k μm^2. The custom designed 4096 bit SA PUF and two SRAM arrays (both 64x64 column-mux 16 arrays) take 8.1% (9.7k μm^2) and 13.0% (15.4k μm^2) of the overall area respectively. The remaining 78.9% of area is synthesized logic. The synthesized logic is composed of the key-generator logic (IO-Ctrl and Test Ctrl) and the AES block. The key-generator logic fits in a total of 3963 GEs and in an area of 24k μm^2. The total area of the key generator (SA/SRAM arrays and control/IO logic) is 49.1k μm^2. The AES was built using 10487 GEs in an area of 34.4k μm^2. The total on-die area of the synthesized logic is more than the sum of the area of individual synthesized blocks because of placement and routing overhead as well as because of many scan-flops that are implemented for back-door access to the SRAM and AES internals (for test purposes) which will not be a part of a final implementation.

5.3 PSS in SRAM and delay-based PUFs

A similar "soft"-information based efficient estimation of reliability should be possible even for the SRAM PUF but would require circuits to control the voltage of bitlines at power-up. RO PUFs have been shown to use frequency measures as the "soft"-information to estimate reliability (as discussed earlier in Section 5). For the arbiter PUFs, where the response depends on the electrical characteristics of many devices, a similar estimation would require a measurement of properties of all those devices, followed by post-processing which may have high overheads. A more efficient method would be multiple evaluations across voltage and temperature, and choosing only the configurations that generate consistent response [10]. This, as we have pointed out earlier in Section 5, may result in an unacceptable increase in tester time and still may not provide any statistical insights about the safety margin in the chosen elements.

5.4 Discussion of Results

We have shown that PSS is an effective way to reduce the errors in the PUF response. The bit error rate (ber) achieved is $< 5 * 10^{-9}$. This is equivalent to a 128-bit key failure rate $< 10^{-6}$ which is the typical targeted failure rate for ECC implementations.

This is how PSS implementation compares to the conventional error correction techniques on metrics discussed in Section 3.7.1.

- For the ΔV_{IN} that provides a ber $< 5 * 10^{-9}$, the percentage of selected bits is $\sim$30%. Hence, the PSS technique on SA PUFs requires $\sim$3.3x SA bits for the generation of 1 reliable bit. The current implementation is capable of generating $>$1200 reliable bits from a pool of 4096 bits. Generation of 128 bits (or 171 bits if the secrecy rate of 0.75 is assumed) would result in a much smaller SA and SRAM array (570 bits for the generation of 171 bits) and smaller peripheral logic.
- The reliability bitmap generated is equivalent to the helper data of conventional ECC techniques in some respects. It is generated at the time of enrollment, stored off-die possibly in the clear, and needs to be imported on-die for generation of the reliable bits. The reliability

bitmap, however, carries no information about polarity of the bits but only the physical location of the potentially more reliable bits. Hence they do not leak any information about the bits unless there was any correlation found in the bits generated from the SA PUFs in the array. However, the biggest contributor of V_{OFFSET} (and hence the polarity of bit) is *local* random variations in the devices of a SA (e.g., random dopant fluctuations and line edge roughness) and hence the bits of the array can be assumed to be largely independent.

- Our bit generator would take $\sim$300 cycles to generate the key and run at high speeds (250 MHz as seen in our 65nm simulations).
- The key generator is able to generate bits with even higher reliability by rejecting more of the raw bits (see Section 5.2.4).

5.4.1 Strong PUF

We describe the framework of a strong PUF realization that was fabricated on our Generation II testchip. The strong PUF leverages the security guarantees of an Advanced Encryption Standard (AES), a standard cryptographic primitive and the reliability of the bits generated by the key generator to realize a secure and reliable strong-PUF. Our proof-of-concept implementation of AES can be optimized from the current $>$10.5k GE to $\sim$2.4k GE [50]. In fact, the reliable bits can be used with any standard keyed one-way cryptographic primitive to realize a strong-PUF.

5.5 Summary of Chapter

In this chapter, we described the post-silicon selection (PSS) technique to improve the reliability of a PUF. We propose the use of the sense amplifiers offset voltage measurement as a metric to pre-characterize the reliability of sense amplifier PUFs. We then present the design of a highly reliable, self-contained, efficient key generator based on PSS of SA PUF elements. Measurements from our 65nm Generation II testchip show that our key generator operates with a bit error rate $BER < 5 * 10^{-9}$, equivalent to a 128-bit key error rate of $< 10^{-6}$. Such low BER is conventionally only achievable using powerful error correction codes (ECC). We leverage this high reliability of the key generator, in conjunction with an AES cryptographic primitive, to build a reliable, efficient, and secure strong PUF.

Chapter 6

Intrinsic Reliability Enhancement by Directed Accelerated Aging (DAA)

As mentioned earlier, we categorize a reliability enhancement technique as *intrinsic* if it makes a fundamental change to the set of PUF core bits. In this chapter we discuss the second intrinsic technique for reliability enhancement – Directed Accelerated Aging (DAA) – which results in a permanent change to the properties of the PUF core bits.

Integrated circuits suffer from a number of aging phenomena that can affect device performance and characteristics over time. Among these, the most prevalent are negative bias temperature instability (NBTI), hot carrier injection (HCI), time dependent dielectric breakdown (TDDB), and electromigration (EM). These aging phenomena can generally degrade the circuit performance and also cause memory remanence in even supposedly volatile memory structures such as SRAM and DRAM [22]. Previous work has evaluated how aging may reduce the PUF reliability (similar to how it would degrade the performance of any regular circuit) [60], but to our knowledge, this is the first work to propose purposely using an aging phenomenon in a directed manner to enhance the PUF reliability.

DAA uses an IC stress phenomenon (e.g., HCI, NBTI) in a directed manner such that it creates a permanent and desired change in the device(s) of the PUF core. Using measurements from our Generation I and Generation II testchips (Appendix A), we are able to demonstrate that both NBTI and HCI based DAA can increase PUF reliability. Measured results show that

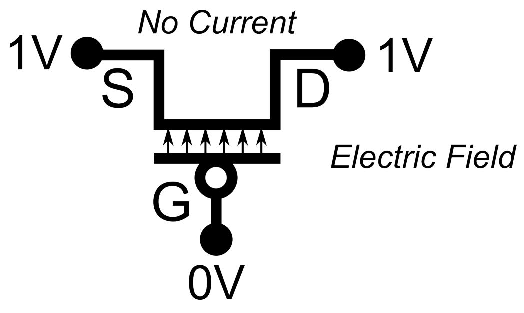
HCI-based DAA can generate response bits with a bit error rate $<5*10^{-9}$. This is equivalent to a 128-bit key failure rate $<10^{-6}$ which is a typical targeted failure rate for error correction code (ECC) implementations [11, 20, 37, 40, 44].

We propose the use of NBTI-based DAA in SRAM PUF cores and HCI-based DAA in sense amplifier PUF cores. As we shall see later in this section, NBTI-based DAA (or NBTI-DAA) is able to reduce errors by only $\sim$40%. The HCI-based DAA (or HCI-DAA) shows a $\sim$100% reduction in errors. Apart from efficacy, there are several other advantages of HCI-DAA over NBTI-DAA as discussed later in Section 6.4. But for completeness, we discuss both the proposals. First, we discuss the aging phenomena - NBTI and HCI.

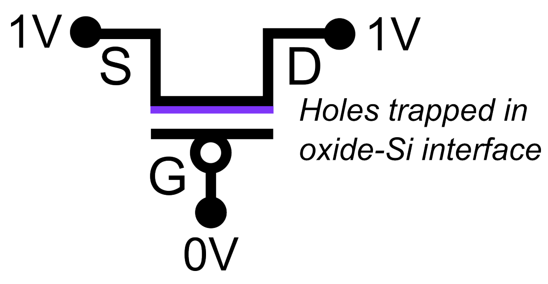
6.1 Aging Phenomena used for DAA

6.1.1 Negative Bias Temperature Instability (NBTI)

NBTI is a dominant failure mechanism for PMOS devices in modern nanometer IC technologies [70]. NBTI results in an increase in the threshold voltage (V_{TH}) of PMOS devices that are subjected to a negative gate-to-source (V_{GS}) biasing over a long duration (Figure 6.1). The exact mechanism of NBTI degradation is not well understood, however, it is known that the V_{TH} degradation has both a dynamic (temporary) and a static (permanent) component. Studies have shown that the permanent shift in V_{TH} because of NBTI is typically in the range $\sim$10–40mV.



(a) PMOS device with negative V_{GS}



(b) Holes trapped in gate oxide over time

Figure 6.1: NBTI phenomenon in PMOS devices

6.1.2 Hot Carrier Injection (HCI)

Figure 6.2 gives an overview of the HCI phenomenon for an NMOS transistor. Figure 6.2(a) shows an NMOS transistor under normal biasing. The gate-to-source voltage (V_{GS}) and the drain-to-source voltage (V_{DS}) are at nominal supply voltage (assumed to be 1V) and the transistor operates in saturation. As V_{DS} increases, as shown in Figure 6.2(b), velocity saturation occurs and for today's short channels, it can occur for much of the channel. Electrons moving at saturation velocity continue to acquire kinetic energy, but their velocity is randomized by excessive collisions such that their average velocity along the field direction no longer increases but their random kinetic energy does. These high energy electrons are called hot carriers and their population increases for higher V_{DS}. A small fraction of these hot carriers acquire enough energy to overcome the silicon-oxide barrier energy and get injected into the gate oxide (the brown square in Figure 6.2(b)).

Transistors with carriers trapped in the oxide require a higher V_{GS} for inversion, effectively increasing their V$_{TH}$. When this stressed NMOS transistor, with trapped electrons, is used under normal V_{DS} biasing as shown in Figure 6.2(c), the NMOS transistor behaves asymmetrically under the two source-drain biasing directions. When the current flows in the same direction under normal biasing, as in the stressed biasing, such that the trapped electrons are near the drain, the NMOS transistor sees only a slight increase in V$_{TH}$. However, when used with the source-drain directionality reversed, such that the trapped electrons are near the source, the NMOS transistor will see a much higher increase in V$_{TH}$. This is because for inversion, most of the charge accumulates in the channel near the source and with trapped carriers near the source, it requires a larger V_{GS} to attract electrons for inversion. Since the electrons are trapped deep into the oxide, most of the increase in V$_{TH}$ is permanent, making HCI an attractive mechanism to reinforce the PUF response.

6.2 DAA using NBTI in SRAM PUFs (NBTI-DAA)

Figure 6.3 shows a typical 6T SRAM cell. Under prolonged storage of the same bit value, V$_{TH}$ of the two PMOS devices in a 6T SRAM cell undergo a differential shift due to NBTI [6]. When

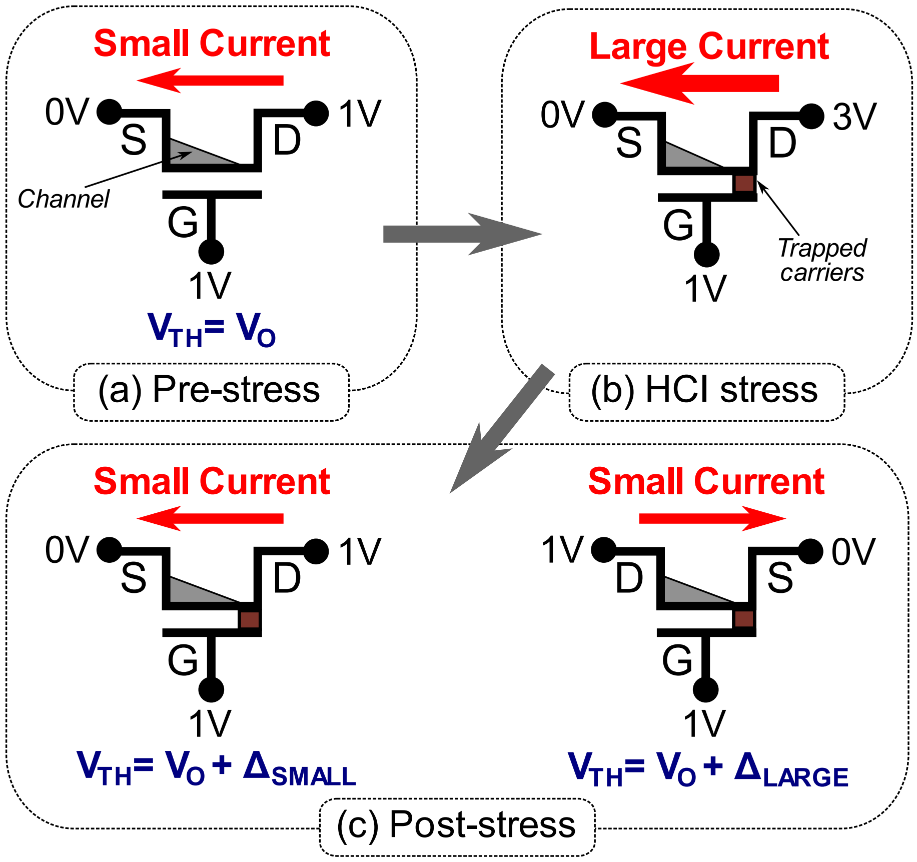


Figure 6.2: (a) Pre-stress NMOS transistor with normal biasing. (b) NMOS transistor under HCI stress conditions. A high V_{DS} generates a large current resulting in some hot electrons getting injected deep into the gate oxide (shown as the brown square). (c) After HCI stress, when the NMOS transistor is biased normally, it sees an increased threshold voltage (V_{TH}). The increase is significant (>100mV) when current is in the opposite direction as during the stress conditions. The increase in V_{TH}, however, is small when current flows is the same direction as during the stress conditions.

a 6T cell is in equilibrium, only one of the PMOS devices is turned on and subjected to negative gate to source bias. V_{TH} of the on device increases due to NBTI whereas the V_{TH} of the other PMOS device does not change. This effect can be leveraged to reinforce a preferred value in bi-stable PUF elements since the polarity of the resolved bit is a strong function of the relative V_{TH} of the devices. When the opposite of the preferred value is stored in the cell, the PMOS device with higher V_{TH} is turned on and it gets weaker because of NBTI degradation. Therefore, the V_{TH} difference between the PMOS devices increases which in return makes the cell more reliable. Accelerated aging can be induced by subjecting the circuit to an elevated temperature and VDD.

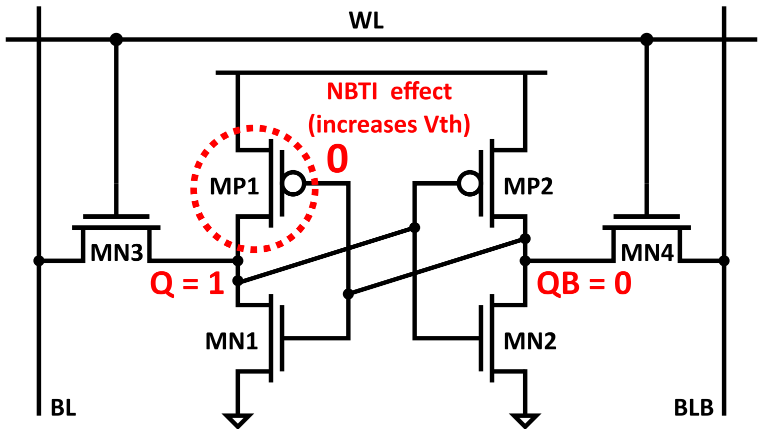


Figure 6.3: How NBTI affects a 6T SRAM cell. If '1' is stored in the SRAM (Q=1 and QB=0), then the gate of the PMOS transistor MP1 is subjected to negative gate to source bias. This results in a slow increase in the threshold voltage of the device because of NBTI. Hence, if the cell stores a '1' for a long duration, MP1 becomes weaker over time. If the cell is powered-up after this effect has taken place, it is less likely to power-up to a '1' than it was before this effect had impacted the PMOS characteristics.

6.2.1 Example Procedure

To use NBTI-DAA in a SRAM PUF, the procedure for reliability reinforcing would be as follows:

- Determine the preferred state of each SRAM cell. For example, an SRAM PUF could be read out multiple times, and each bit assigned a probability based on how often it powered-up to 1.

- Set the state of each SRAM cell to reinforce the preferred value of the cell. As explained earlier, this would require writing the opposite value of their preferred state to reinforce their preferred state.
- Stress the SRAM PUF to induce accelerated aging. This typically involves increasing the temperature of the die using a burn-in oven or temperature controlled chamber and increasing the supply voltage.
- Return to normal PUF operating conditions.

6.2.2 Reliability Enhancement: Experimental Results

To evaluate the efficacy of NBTI-DAA, the testchip SRAM arrays were subjected to directed burn in stress for a total of 120 hours when data 0 was written to all memory locations, which should bias the cells to resolve to the 1 state. The applied stress conditions were ambient temperature of 100°C and supply voltage of 1.3V (the nominal supply voltage is 1.2V). Before burn-in, we read and record the 1000 power-up values of every cell in the arrays, and repeat this data collection after multiple stress times ranging from 6 hours to 120 hours. Figure 6.4 shows how 120 hours of NBTI-DAA reinforces the cells that showed a preferential bias towards 1. Since the SRAM cells are symmetric, the same behavior is seen when stressing the cells with data 1 (i.e., biasing the cells towards the 0 state). Figure 6.5 shows the percent errors as a function of number of hours of burn-in aging. Across 120 hours of directed aging, the unreliable bit percentage was reduced from 2.5% to under 1.5%, a reduction of over 40% in bit errors due to DAA.

6.2.3 NBTI-DAA as an Attack

Just like NBTI-DAA can be used to increase the reliability of a PUF, it can also be maliciously used to decrease the reliability by aging the cells in the opposite direction as their preferred value. This possibility underscore the importance of not allowing users the ability to write to SRAMs used as PUFs. A malicious user could determine the preferred power-up value of a SRAM PUF, and if allowed write access, could use NBTI-DAA to reduce the PUF reliability or in an extreme case completely alter the response values.

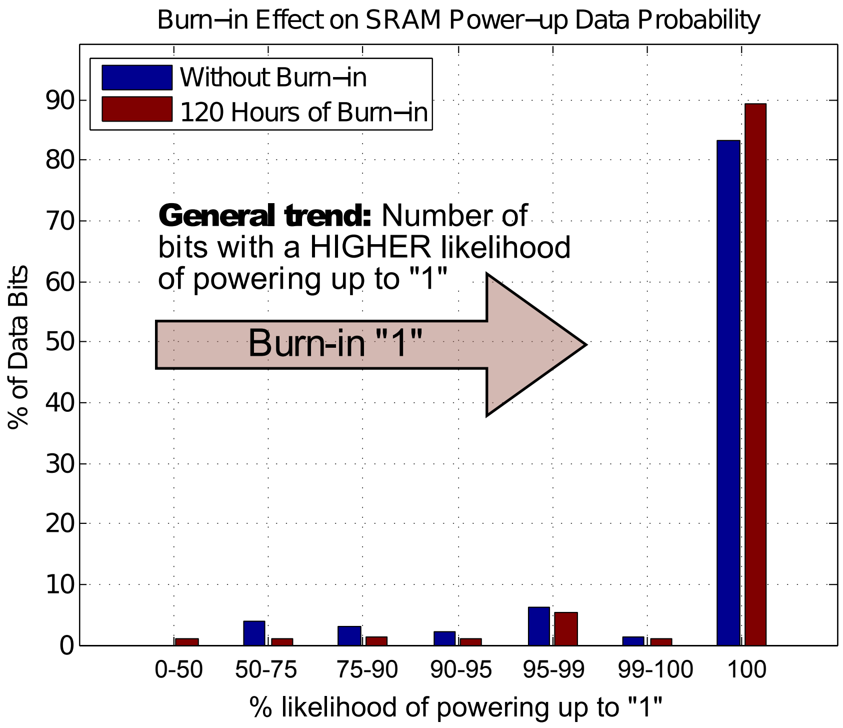


Figure 6.4: Biasing the cells toward 1 via NBTI-DAA. The empirical probability of a bit powering up to a '1' is measured over 1000 power-ups before and after directed aging for 120 hours at 100°C and 1.3V VDD which is equivalent to 0.8 years of aging under nominal conditions (1.2V and 27°C). Of all the bits that were considered '1' (majority voting in 1000 power-ups) before aging, the % of bits that powered up to '1' in all of 1000 power-up measurements (100% reliable) increased from 83.3% to 89.3% by NBTI-DAA.

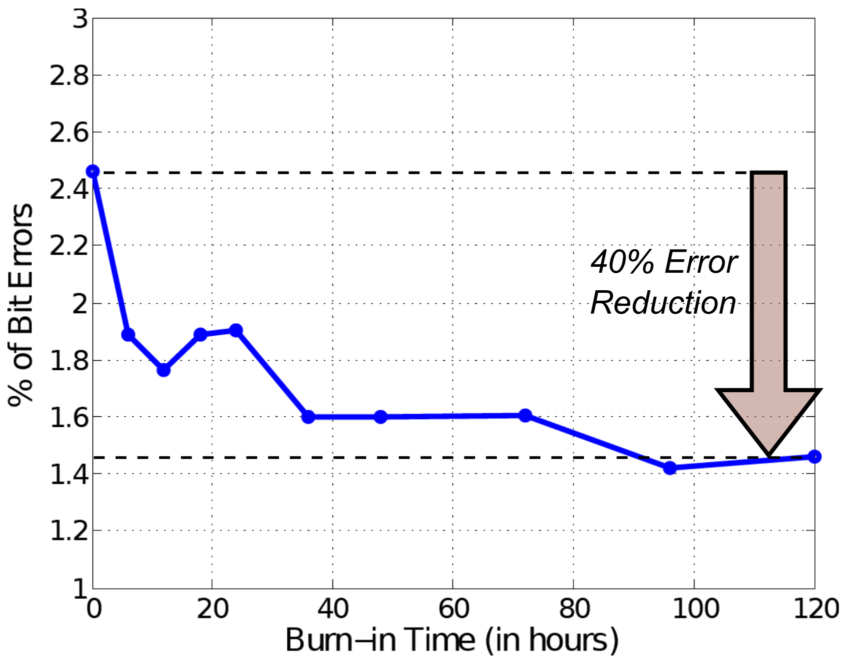


Figure 6.5: Reduction in percent errors with NBTI-DAA burn-in time for one of the testchips. Errors reduced from 2.5% to under 1.5% with DAA for 120 hours at 100°C and 1.3V power supply which is equivalent to 0.8 years of aging under nominal conditions (27°C and 1.2V).

6.2.4 Implementation Options

While control of this procedure can be performed outside of the chip, this would require the chip to be able to output its PUF core bits as well as accept external data inputs to be written into the PUF core. These output and input access points must be disabled before the chip is sent into the field via fuse, laser, focused ion beam (FIB), or similar destructive measures. Even so, such paths may be recoverable by an attacker, as researchers have done with commercial microprocessor test access port scan chains [35].

Alternately, we could implement value reinforcement using a built-in self-test (BIST) block. This block would orchestrate the reinforcement procedure described above, but since it would be on-die, the PUF core bits would not need to be exported off-die. This would dramatically increasing the difficulty for an attacker to access those bits. The BIST block would need to contain a small amount of sequencing logic and a memory large enough to hold the soft information for the entire PUF core. For each PUF core bit, the BIST block would need to store a number of bits of soft information depending on the desired resolution. Since this would require a large amount of memory, we could reinforce the PUF in stages, thus reducing the amount of needed memory by a factor equal to the number of stages. Additionally, this could be shared with the multiple evaluation memory as discussed in Section 4.1.

6.3 DAA using HCI in SA PUFs (HCI-DAA)

In this section we describe a PUF circuit that uses HCI-based DAA to increase reliability. The PUF is based on a StrongARM type sense amplifiers as the core element (as described in Section 2.3.2. Figure 2.9 shows a StrongARM sense amplifier circuit topology, which we use as the basis of our PUF. Under ideal conditions, an ideal SA would correctly amplify even the smallest of input differential voltages. In practice, however, variations in the devices of an SA may result in an offset (or bias), a measure of the natural tendency of the SA to resolve to a particular polarity. To ensure correct operation, the SA inputs need to have a difference larger than the offset.

To use as a PUF core, SA inputs (BL and BLB in Figure 2.9) are shorted together (i.e., set to the same voltage, zero differential input) and the SA is fired. The SA will then resolve to a value determined by its individual offset [8]. The offset of the StrongARM SA (Figure 2.9) is a strong function of the difference in V_{TH} of devices `MN1` and `MN2`. The offset (and hence the reliability) of the SA can be increased by increasing the difference in the V_{TH} of `MN1` and `MN2`.

As we have seen in the previous chapter, the offset of a SA is strong indicator of its reliability as a PUF [8, 9]. A SA with high offset (i.e., a strong bias to resolve to a particular polarity) will likely resolve to the same polarity across environmental variations and over aging. The exact offset cutoff for such high reliability may vary with technology, design specifics, or measurement conditions but the key idea here is that use of SAs with higher offsets will result in PUFs of higher reliability.

6.3.1 HCI-Enabled Sense Amplifier (HCI-SA) PUF Structure

We modified the StrongARM SA described above to enable the use of HCI to reinforce the individual offset of each SA (Figure 6.6 and Figure 6.7). The measurement of the offset polarity and reinforcement is done completely autonomously by the circuit. The basic StrongARM SA is shown in black, while the additionally HCI reinforcement circuitry is shown in blue.

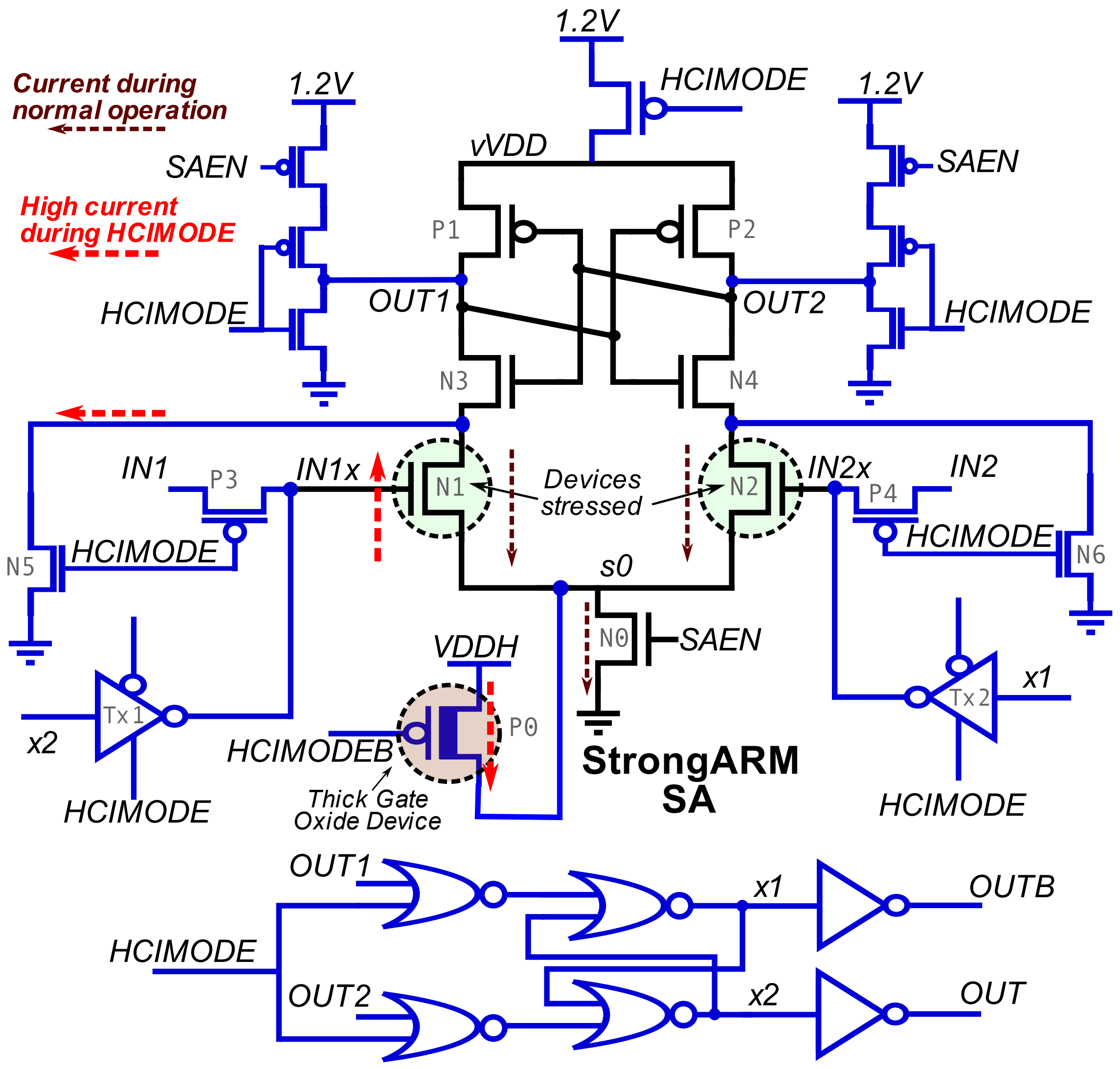


Figure 6.6: HCI-enabled sense amplifier PUF (HCI-SAPUF) core schematic. The baseline StrongARM sense amplifier (shown in black) has been modified with peripheral circuits (blue) to enable HCI aging. Note the use of high oxide device to interface with the high 3V *VDDH*.

Post-manufacturing, and before first use as a PUF, the HCI-SAs can be programmed for

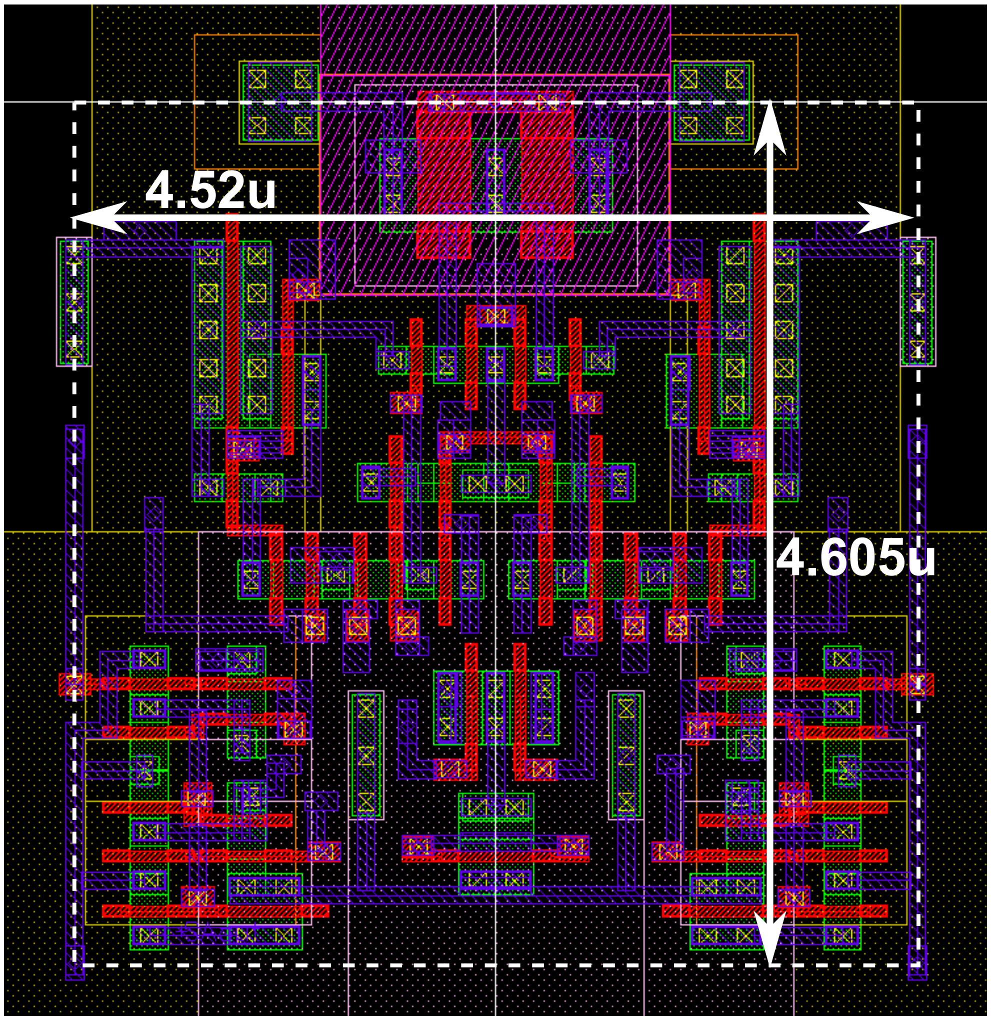


Figure 6.7: HCI-SA PUF was laid out symmetrically in an area of 20.8μm^2 (4.52μmx4.605μm).

higher reliability by stressing either of N1 or N2 (Figure 6.6). This is done individually for each SA, but since the offset reinforcement circuitry is self-contained for each SA, all SAs are reinforced in parallel. If *offset*$_0$ is the offset before stress, then the offset after stress (*offset*$_S$) should have the same sign as *offset*$_0$ and a higher magnitude.

The HCI-SAs operates in two modes, *normal mode* and *HCI mode*, controlled by the signal *HCIMODE*. In the *normal mode* (*HCIMODE*=0), the HCI-SAs act as normal StrongARM SAs. In the *HCI mode* (*HCIMODE*=1), one of N1 or N2 is stressed.

The reinforcement of offset is done in two steps: 1) offset polarity measurement and storage, and 2) HCI offset reinforcement.

Offset polarity measurement and storage. Which of N1 or N2 is stressed is determined by nodes *x1* and *x2*. The values of *x1* and *x2* are internally generated *normal mode* during the first step of offset reinforcement (Figure 6.8). For example, if V$_{TH}$ of N1 is higher than V$_{TH}$ of N2 for a particular HCI-SA, and all other devices are matched, then in this step, the HCI-SA, when fired, will resolve to a 1 (*OUT1*=1,*OUT2*=0) and this value is latched as *x1*=1, *x2*=0.

HCI offset reinforcement. During the second step of offset reinforcement, *HCIMODE*=1 and P3 and P4 are disabled and the tri-state buffers Tx1 and Tx2 are enabled. The values at *x1* and *x2* force *IN1x*=1 and *IN2x*=0. This is followed by a pulse of high voltage (∼3V) at *VDDH* resulting in a high current path through P0, N1, and N5. The devices are sized such that most of the voltage drop (∼2.5-2.8V when provided with a pulse of 3V) is seen across the drain-source of N1. This creates the stress conditions for N1 as described earlier in Section 6.1.2 and results in an increased V$_{TH}$ of N1 when the SA is used later as a PUF in the *normal mode*. Note that the high voltage (*VDDH*) is connected only to a single thick gate oxide PMOS device (P0) per HCI-SA. A thick gate oxide device can withstand a higher V_{GS} without gate oxide breakdown and is a common process technology option since thick oxide devices are needed in the pads. The amount of stress (i.e., the extent of offset reinforcement) is controlled by the pulse width and the voltage of *VDDH*. A supply of ∼2.5-3V should be readily available on die as the I/O pad supply and hence the offset reinforcement step does not require a separate dedicated voltage supply.

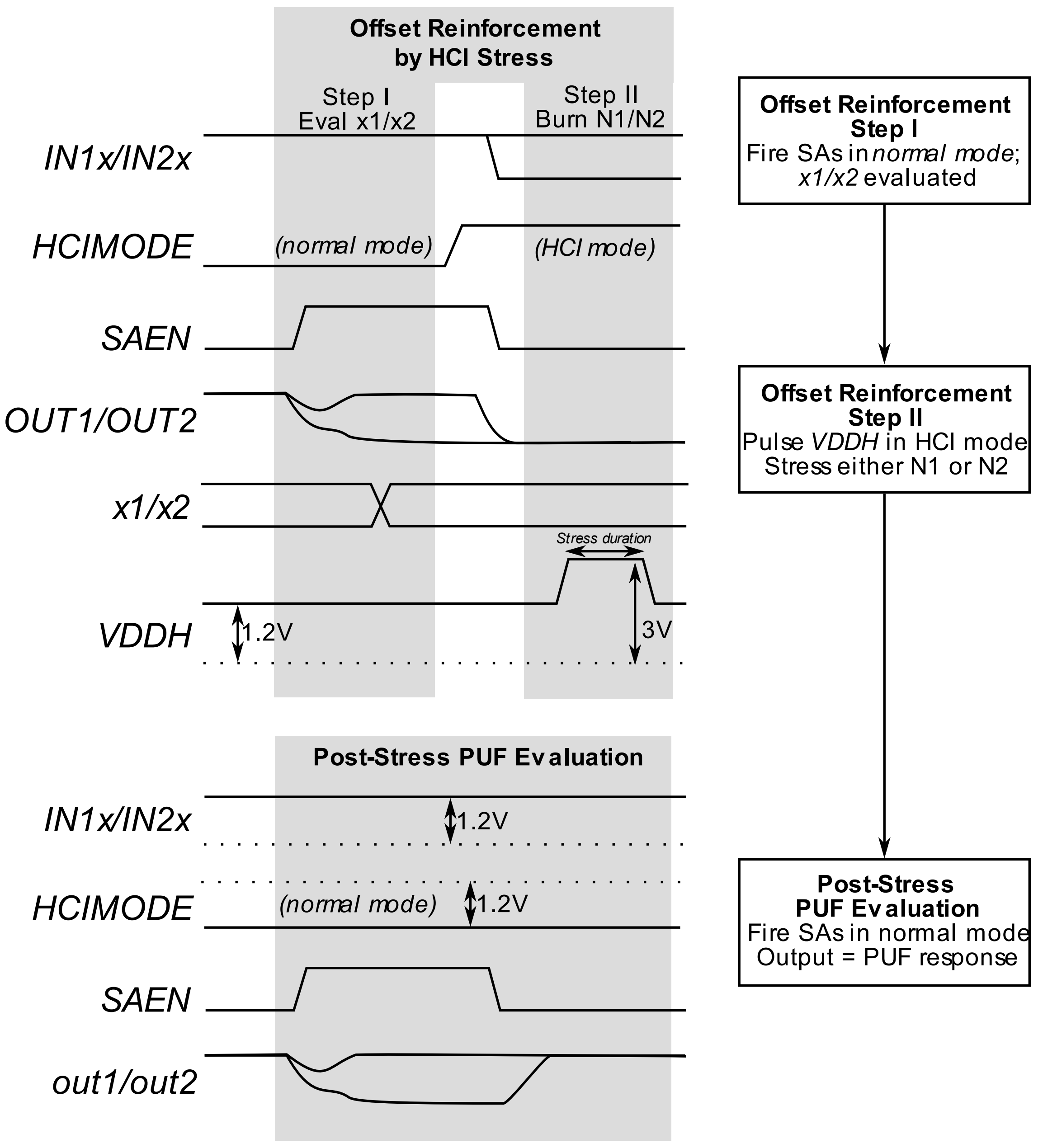


Figure 6.8: Steps of operation of HCI-SA.

6.3.2 Use case

The proposed use case for the HCI-SA PUF is for it to undergo a one-time HCI reinforcement step immediately post-manufacturing as detailed earlier in this section. This one-time step requires a few tens of seconds of HCI stress resulting in a permanent offset shift in each of the HCI-SA PUF core circuits in the direction determined by random process variations. After this one-time stress, the HCI-SA PUF is used just like a regular SA PUF by activating it in the *normal mode*. HCI is only applied to the devices at this initial reinforcement, and devices in the field are not subjected to any additional HCI stress.

6.3.3 Shift in Offset by HCI Stress: Experimental Results

Using the testchip implementation as explained later in Section A.2, we measured the HCI-SA PUF element offset, reliability across environmental variations and aging, uniqueness, and randomness. The measurements are taken across a voltage range of $\pm 20\%$ of the nominal 1V VDD (i.e., 0.8V to 1.2V) and temperatures of -20°C, 27°C, and 85°C. We used a TestEquity Model 107 temperature chamber to fully enclose the test PCB during temperature variation testing.

The HCI-SAs were incrementally stressed using 3V *VDDH* pulses of width 1s, 4s, 20s, and 100s which resulted in a cumulative stress of 1s, 5s, 25s, and 125s. As mentioned earlier, the 3V supply is connected only to a thick gate oxide PMOS transistor per HCI-SA to avoid oxide breakdown of other devices in the circuit. Offset reinforcement only requires firing the HCI-SAs once (all fire together, since *SAEN*, *IN1*, and *IN2* are shared across all HCI-SAs in an array) in the *normal mode* followed by a pulse of high voltage at *VDDH* in *HCI mode*.

Offsets before and after stress. To evaluate the efficacy of HCI in altering the SA offset, we measure the offset of each of the 1600 HCI-SAs before and after HCI offset reinforcement. To measure the offset (either before or after HCI stress), the input differential (i.e., the voltage difference between *IN1* and *IN2*) is swept from -400mV to 400mV in steps of 10mV. At each step, the HCI-SAs are fired multiple times. The outputs after every activation are read out of the output scan chain. These are then post-processed to measure the offset of each HCI-SA in the array. Figure 6.9 shows the measured offset of all HCI-SAs of a chip, before and after HCI stress, when arranged in order of their *offset*$_0$ in ascending order. As expected, the shift in offset is higher for longer stress durations. For a 1s stress, the shift in offset is $\sim 10 - 50mV$ and for a stress of 125s, the shift is $\sim 150 - 300mV$.

Figure 6.10 shows the scatter plot of the magnitude of the shift in offset ($|\Delta_{offset}|$) vs.|*offset*$_0$| for different stress durations. As expected, $|\Delta_{offset}|$ is higher for longer stress durations. For a 1s stress, $|\Delta_{offset}|$ is $\sim 10-50mV$ and for a stress of 125s, $|\Delta_{offset}|$ is $\sim 150-300mV$. Moreover, on average (bold lines in Figure 6.10), $|\Delta_{offset}|$ is slightly larger for SAs with low magnitude of *offset*$_0$, which is desirable since SAs with low magnitude of offset would need a larger shift for reliability.

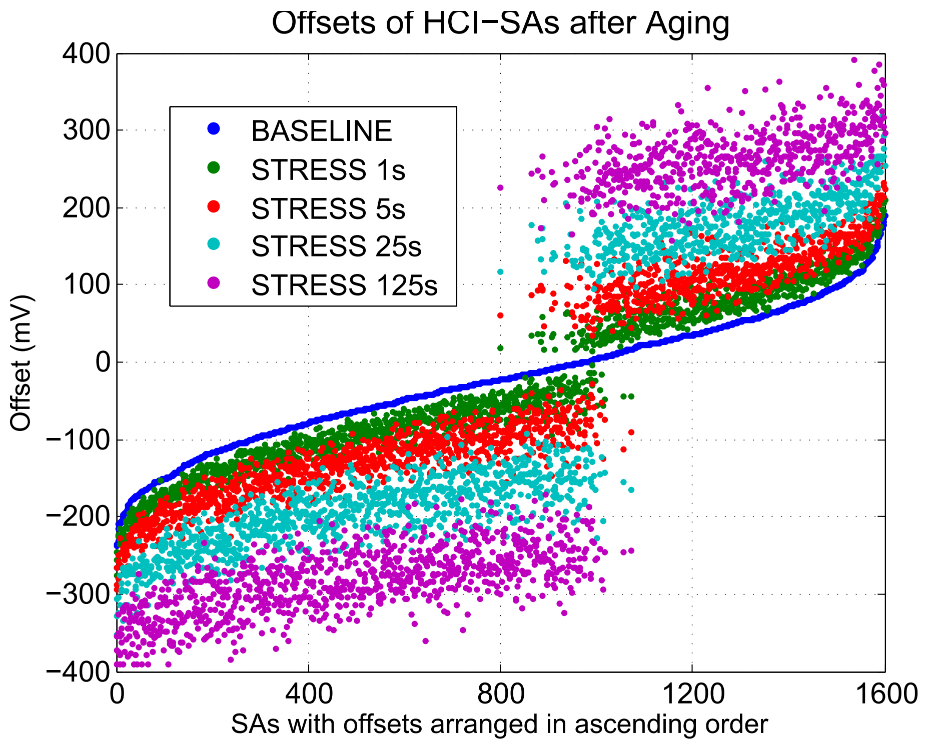


Figure 6.9: Measured offset of all 1600 HCI-SAs on a die before and after HCI stress. The SAs are arranged in order of their offset before stress in ascending order. The minimum magnitude of offset after stress of 1s, 5s, 25s, and 125s was found to be 5mV, 28mV, 92mV, and 158mV respectively.

Figure 6.11 shows the measured distribution of the offset for one chip before and after HCI stress at 1.0V and 27°C. The *offset*$_0$ has a typical normal spread as expected. The mean is slightly skewed ($\mu_0 = -23mV$) and may be due to a layout systematic bias. The standard deviation of

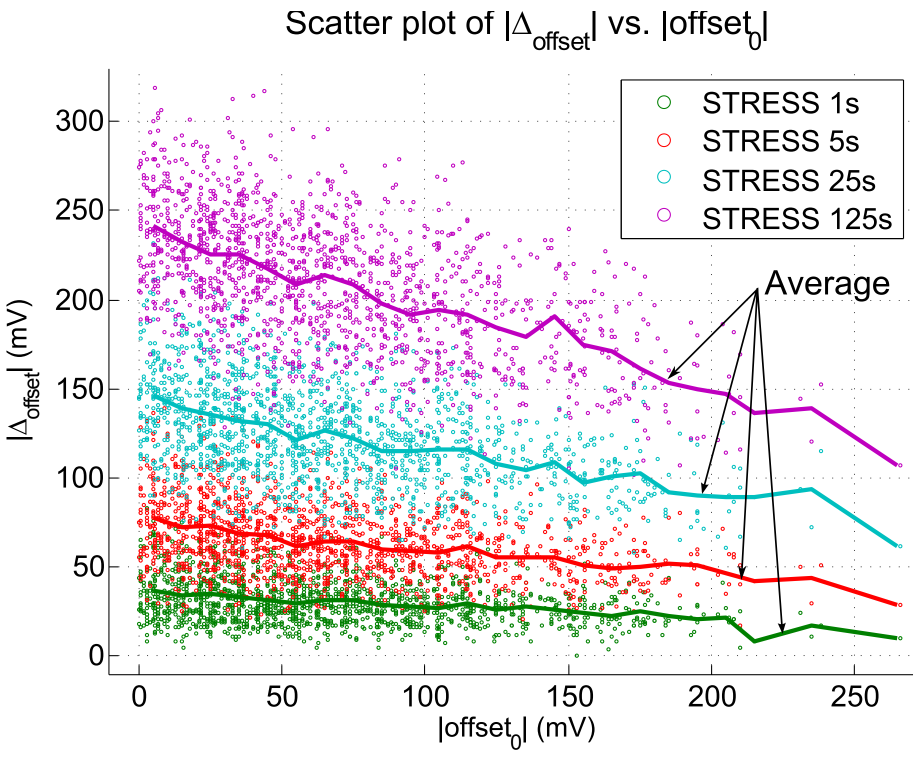


Figure 6.10: Scatter plot of shift in measured offset after HCI stress vs. the measured offset before stress. The average shift in offset is plotted in bold lines.

offset$_0$ is 79mV. After stress, we see that the distribution splits into two groups. SAs with negative *offset*$_0$ have their offset shifted to the left (more negative) and SAs with positive *offset*$_0$ have their offset shifted to the right (more positive). The minimum magnitude of offset after stress of 1s, 5s, 25s, and 125s was found to be 5mV, 28mV, 92mV, and 158mV respectively.

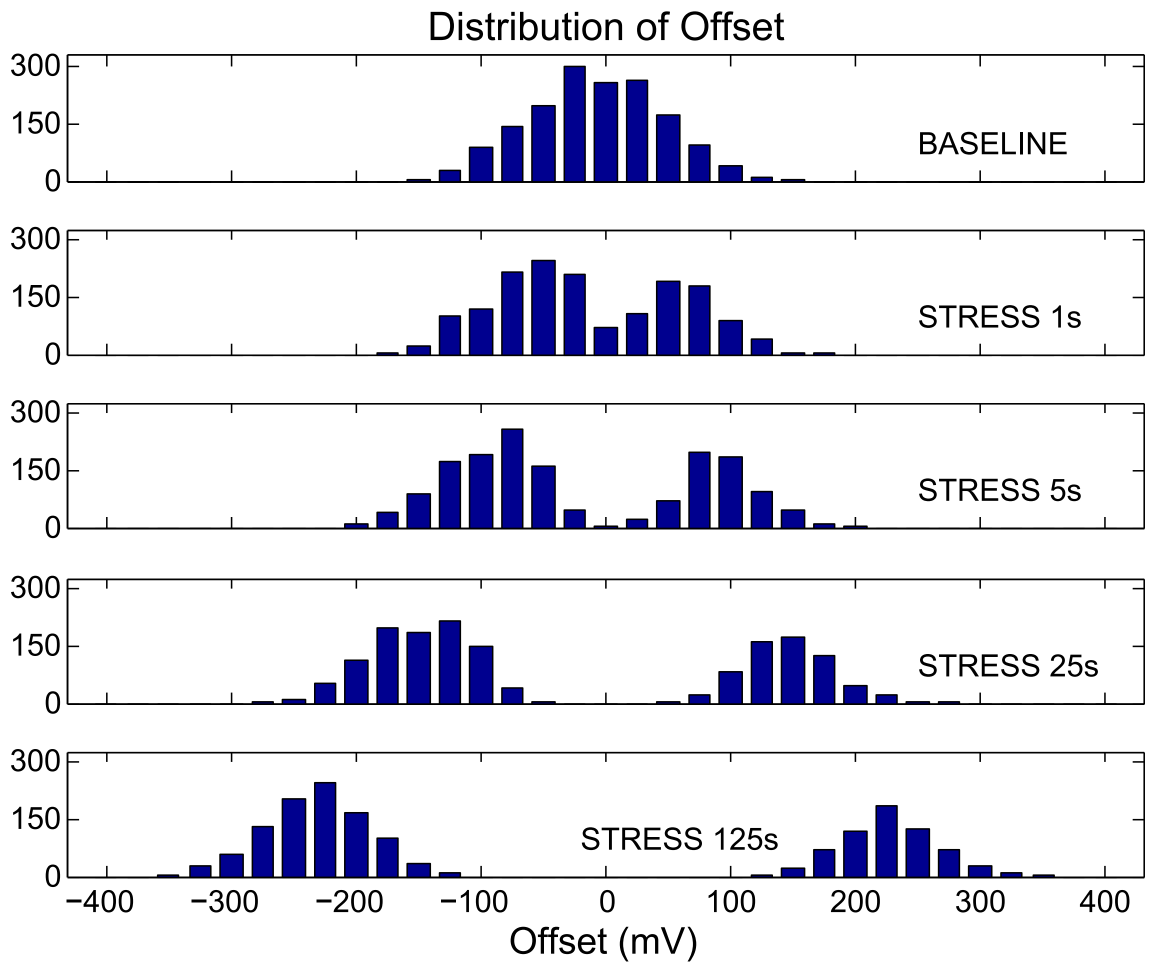


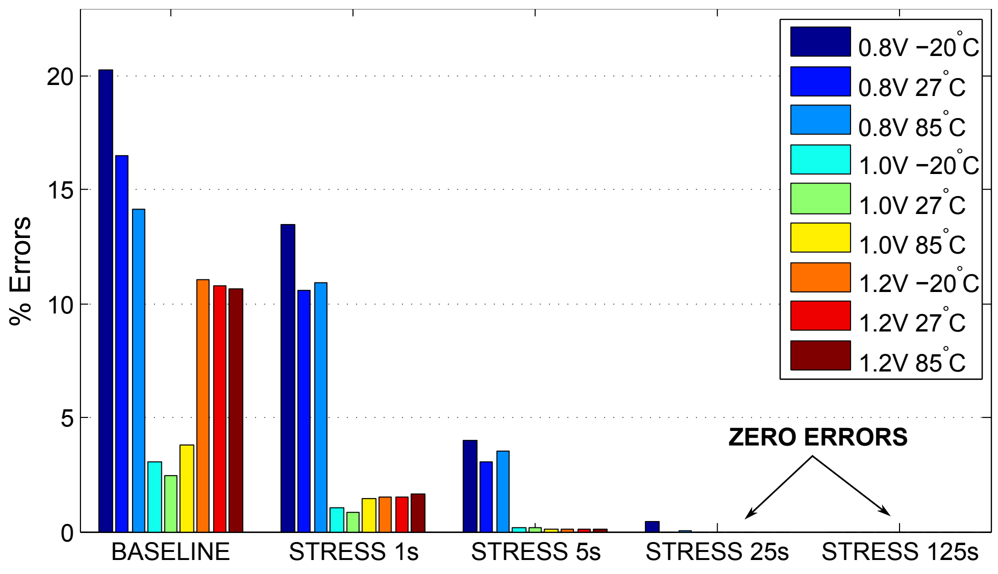
Figure 6.11: Distribution of measured offset of all of 1600 self-programmable HCI-SAs on a die before and after HCI stress (measured at 1.2V and 27°C). After stress, we see that the distribution splits into two groups. SAs with negative *offset*$_0$ have their offset shifted to the left (more negative) and SAs with positive *offset*$_0$ have their offset shifted to the right (more positive).

6.3.4 Reliability Enhancement: Experimental Results

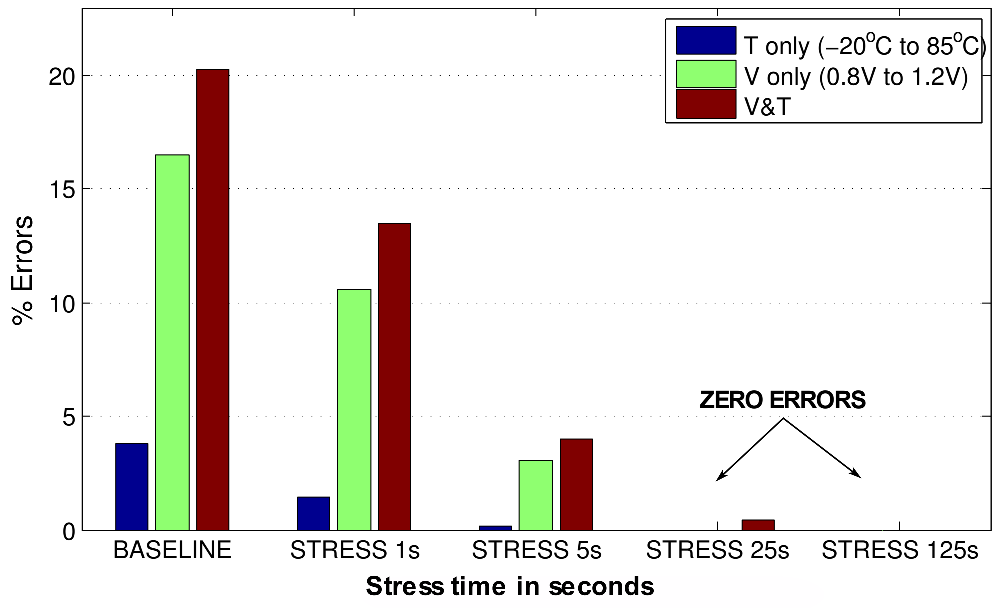
Figures 6.9 and Figure 6.11 show that HCI stress can increase offset in SA. However, offset is an indirect measure of reliability and PUF reliability can be directly measured by multiple evaluations across environmental variations and over aging. For direct measurement of reliability, we first perform small-scale measurements to identify the worst-case corner for reliability and then perform large-scale measurements on the nominal and the worst case corners.

Small-scale measurements. For small-scale measurements for reliability across environmental variation we do the following.

1. Perform 100 PUF evaluations at each possible combination of voltage (0.8V, 1.0V, 1.2V) and temperature (-20°C, 27°C, 85°C). Each PUF evaluation generates 1600 response bits corresponding to the 1600 HCI-SA elements in the array. The majority vote of the 100 responses at the nominal conditions (27°C and 1.0V) is considered the golden response against which the response at other conditions will be compared.
2. At every combination of voltage and temperature, each of the 100 evaluations is compared to the golden response. We define $Error_{i,V_1,T_1}$ as the number of bits out of the 1600 HCI-SA outputs that do not match the golden response in the i^{th} evaluation at voltage=V_1 V and temperature=T_1°C.
3. We define % errors at a voltage-temperature combination ($Error_{V_1,T_1}$) as the maximum $Error_{i,V_1,T_1}$ across the 100 evaluations. The % errors across voltage-only variations ($Error_{V only}$) is defined as the maximum errors across the 100 evaluations at all voltage variations and at nominal temperature i.e., maximum of $Error_{0.8,27}$, $Error_{1.0,27}$, and $Error_{1.2,27}$. Similarly, % errors across temperature-only variations ($Error_{T only}$) is defined as the maximum errors across the 100 evaluations at all temperature variations and at nominal voltage i.e., maximum of $Error_{1.0,-20}$, $Error_{1.0,27}$, and $Error_{1.0,85}$. The overall % errors ($Error_{V\&T}$) is defined as the maximum errors across the 100 evaluations at all voltage and temperature combinations. Using this methodology, the reported % errors is the largest % of bits that were erroneous for any of the voltage/temperature conditions in any of the 100 PUF evaluations performed at that voltage/temperature. In other words, if we had an ECC that had



(a) Errors across environmental conditions



(b) Overall errors

Figure 6.12: Reliability of HCI-SAPUFs shown as a percentage of errors (100 - % reliability). % Errors shown are the maximum errors across 100 evaluations. (a) Errors across all the environmental conditions. Errors were measured for voltage variations of $\pm 20\%$ from nominal 1V and temperatures of $-20°C$, 27°C, and 85°C. (b) Errors across only voltage, only temperature, and *all* voltage and temperature variations.

the capability to correct that % of the bits, we would have a perfect response (i.e., matching the golden response) every evaluation.

These measures of reliability were taken for a die before and after different stress durations. Figure 6.12 shows the improvement in reliability for SAs with different stress durations and when expressed as % errors. Figure 6.12(b) shows the overall errors (across *all* voltage and temperature variations considered) reduce from 20.3% to 13.5%, 4.0%, 0.43%, and 0% when stressed for 1s, 5s, 25s, and 125s respectively. Note that this means that with 125s stress, there were no errors for any of the 1600 SAs, across all of 100 evaluations at all voltage and temperature combinations. Variations in voltage have a stronger impact on reliability as compared to temperature variations. For temperature only variations, the % errors reduce from 3.8% to 1.4%, 0.19%, 0%, and 0% when stressed for 1s, 5s, 25s, and 125s respectively; and for voltage only variations, the % errors reduce from 16.5% to 10.6%, 3.1%, 0%, and 0% when stressed for 1s, 5s, 25s, and 125s respectively. Figure 6.12(a) shows that highest number of errors are seen at low-temperature and low-voltage (0.8V and $-20°$C).

Large-scale measurements. Once we have identified the worst case corner for reliability as low-temperature and low-voltage (0.8V and $-20°$C), we perform large scale measurements at the corner. We ran 125,000 measurements at both worst case corner and the nominal conditions. As defined earlier in Section 5.2.3, we define our key error rate $KER = 1 - (1 - BER)^{171}$ (where BER is the bit error rate) and the worst case bit error rate (BER_{WC}) as follows. If no errors were found in N runs of S bits, then we conservatively assume that the first error would occur in the very next measurement and define our $BER_{WC} = 1/(N * S + 1)$. Due to the pessimistic definition we can safely claim that for our design the measured BER is at least lower than BER_{WC}. Since no error were observed in any of the 1600 SAs in the 125,000 measurements at worst case, our $BER_{WC} = 5 * 10^{-9}$ and we can claim that experimentally observed $BER < 5 * 10^{-9}$. This is equivalent to a $KER < 10^{-6}$.

6.3.5 Permanence of Reliability Reinforcement

To measure permanence of HCI-stress over aging, we measure the reliability of a HCI-stressed chip before and after aging, simulated in a shorter duration using elevated temperature and voltage. We bake a chip, originally HCI-stressed for 125s, at 150% of nominal 1V (= 1.5V) and $100°C$ for 93 hours, resulting in an Acceleration Factor of 161.4 and hence an aging of $\sim$1.7 years for a chip operating at nominal conditions (1.0V and 27 °C) [39]. Measurements showed 0% errors before and after accelerated aging, suggesting that the impact of HCI-stress is not significantly reversed with aging.

Further, to understand subjectively the impact of aging on reliability reinforcement, we measured the offset of each of the 1600 HCI-SAs immediately after 125s HCI-stress and then after 18 hours and 93 hours of baking the chip in the conditions noted above (Figure 6.13). This is equivalent to an aging of 0.33 years and 1.7 years for a chip operating at nominal conditions. We found that the mean reversal of offset for these aging times was 27.9mV and 35.4mV respectively. This suggests that although some offset reversal happens, there is still a significant percentage of the offset shift that is permanent. Offset reversal would be most concerning for HCI-SAs with lower magnitude of offset before aging. Fortunately, it was observed that the reversal of offset was lower for such HCI-SAs. A possible explanation for this could be that a smaller number of charge traps exist for such HCI-SAs and offset reversal would happen by releasing a percentage of these charge traps. Hence, the offset reversal was much higher for HCI-SAs with larger magnitude of offsets (and hence a higher number of charge traps) but this is of lesser concern, since the resulting offset is still of high magnitude. Also, the measured offsets at two aging times show that of the 35.4mV of the mean offset reversal seen with 1.7 years of aging, nearly 80% happened in the first 0.33 years and only 20% in the next 1.37 years. This suggests that offset reversal saturates quickly and only a small amount of offset reversal can be expected to happen beyond 1.7 years of aging.

6.3.6 Uniqueness and Randomness of HCI-SAs

Uniqueness. Uniqueness is a measure of how uncorrelated the response bits are across chips, and ideally the response bits should differ with a probability of 0.5. The Hamming distance of a k-bit response from ideally unique chips should follow a binomial distribution with parameters $\mathcal{N} = k$ and $p = 0.5$ and the mean of the HD distribution should be equal to k/2. For our case, we create 100 16-bit response words (i.e., k=16) from the measured outputs of the 1600 HCI-SAs on

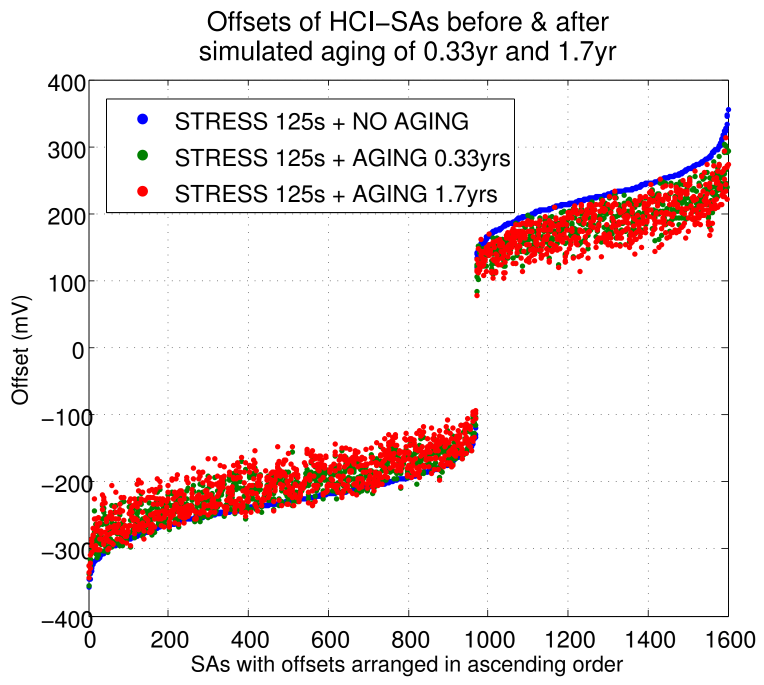


Figure 6.13: Measure of the permanence of reliability reinforcement (offset shift due to HCI-stress). Figure shows the measured offset of all 1600 125s HCI-stressed HCI-SAs before and after simulated aging of 0.33 and 1.7 years.

three chips. These words are generated at 27°C and 1.0V after the HCI-SAs have been stressed for 25s. Figure 6.14 shows that the the pair-wise HD of response bits from the three chips is close to ideal with means of 7.32, 7.36, and 7.50.

Randomness. PUF randomness is a measure of the unpredictability of the response bits. In an ideal random response, the %1's and %0's in the response should be equal. In our measured response of 1600 HCI-SAs from three chips, after a 25s stress, the %1's were found to be 60.6%, 63.6%, and 61.4% which corresponds to entropy of 0.967, 0.946, and 0.962 respectively. The HCI-SAs were designed and laid out symmetrically (Figure 6.6 and Figure 6.7) and we suspect this small but consistent bias across chips to be due to some undesired systematic bias in the layout. We note that our HCI-SAs are based on simple StrongARM SAs and our study of the bias of simple StrongARM SAs have concluded that their uniqueness and randomness characteristics are equal to or better than other PUF types (Section 3.4).

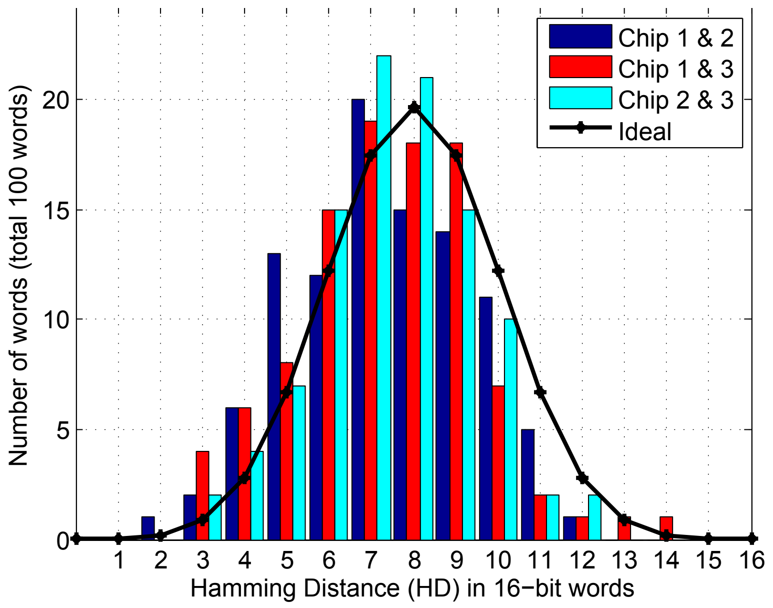


Figure 6.14: Histogram of Hamming distance (HD) of response words from the HCI-SAs across three chips. Also shown is the probability mass function of the HD in responses from ideally unique chips. For the HD comparison, the response bits from 1600 HCI-SAs on a die are grouped to create 100 words of size 16 each. The pair-wise HD of response bits from the three chips is close to ideal with means of 7.32, 7.36, and 7.50.

6.4 Comparing DAA-NBTI and DAA-HCI

We compare the two DAA techniques based on the analysis presented in Section 6.2 and Section 6.3. First, in terms of efficacy, DAA-NBTI in SRAMs is able to achieve only $\sim$40% reduction in response errors. In comparison, DAA-HCI can achieve $\sim$100% reduction in response errors. Hence, DAA-NBTI can at best be seen as a small-scale reliability enhancement solution and would still require ECC blocks to correct errors for application that require high reliability. Results from Section 6.3.4 suggest that DAA-HCI can be used as a stand-alone solution to achieve a desired reliability without any ECC blocks. Second, the response reinforcement in DAA-NBTI requires long baking times (e.g., $>$ 20 hours) that are incompatible with an industrial high-volume-production manufacture and test flow. DAA-HCI requires only a very short (1-2 minutes), one-time reinforcement stress which can be achieved at any time during manufacture test or by the end user. Third, NBTI-stress usually requires elevated voltage and temperature and results in stressing the entire chip. DAA-HCI for HCI-SAs is a very localized phenomenon and impacts only a few thick-gate oxide devices. DAA-NBTI stress, however, can be localized by decoupling the VDD of the SRAM from other VDDs on chip, and applying NBTI stress only by elevating the decoupled SRAM VDD while keeping the temperature at nominal. Fourth, offset reinforcement due to HCI-stress has high permanence as discussed in Section 6.3.5. On the other hand, V_{TH} shifts due to NBTI are not permanent, and transistors return to near their initial characteristics over time [7, 55].

Overall, DAA-HCI is a clear winner compared to DAA-NBTI on several fronts. The only positive for DAA-NBTI is that it does not require a modified cell and can provide a small scale improvement on the traditional 6T SRAM circuits. DAA-HCI application, on the other hand, required a modified StrongARM sense amplifier structure which is $\sim$8.4x larger than the smallest StrongARM sense amplifier structure that we have built. However, these modified cells are capable of parallel, autonomous reinforcement. If we allow a serial and externally controlled reinforcement (similar to what would be required for DAA-NBTI in SRAMs) and sharing of the thick-gate oxide device (the single largest device in the cell), we can reduce the area of the HCI-SA to under $10\mu m^2$ which is 4x the size of the smallest StrongARM sense amplifier.

6.5 DAA in Other PUF Types

The PUF core response depends on the electrical characteristics of much fewer devices in bistable PUF cores compared to the case of a delay based PUF cores. For example, the SRAM and SA PUF core response depends on the electrical characteristics of 4-6 devices. On the other hand, a 7-stage ring oscillator pair that is compared for their frequencies for the generation of a response bit has at least 28 critical devices (one NMOS and one PMOS per stage per ring). Hence, even though it should be possible to apply DAA in delay PUFs, it may have high overheads.

6.6 Discussion of Results

We have shown that DAA-HCI is an effective way to reduce the errors in the PUF response. The bit error rate (ber) achieved is $< 5 * 10^{-9}$. This is equivalent to a 128-bit key failure rate $< 10^{-6}$

which is the typical targeted failure rate for ECC implementations.

This is how DAA-HCI compares to the conventional error correction techniques on the metrics discussed in Section 3.7.1.

- HCI-SA cell proposed is ∼20x the area of a SRAM or ∼10x the area of a SA. However, the core HCI-SA cell has much fewer devices. Many of the devices exist to enable a parallel, autonomous reinforcement of each cell. If we allow a serial and externally controlled reinforcement by removing the offset polarity storage circuitry (20 transistors), and sharing the thick gate oxide device (single largest transistor) across cells, we can reduce the HCI-SA cell to $< 10\mu m^2$ (from the current $20.8\mu m^2$), equivalent to 10x SRAM cell area or 5x SA area.
- No helper data is generated and hence this eliminates all overheads and security vulnerabilities associated with helper data.
- Bit generation takes 1 cycle and can be achieved in ∼ access time of a SRAM (<1 ns for HCI-SAs in 65nm from our simulations).
- The reinforcement is a one-time step. This step is self-contained, does not require connection to an IC tester, and only requires that the chip be powered. It does not require any additional power supply and uses the available I/O pad supply for reinforcement. Reinforcement can be done at any time during manufacture test or even in-the-field after deployment. The required HCI stress time of 1-2 minutes, and could be done during IC burn-in or in-system-test, both of which are typically longer than the time needed for HCI stressing.

6.7 Summary of Chapter

We presented novel techniques to improve the reliability of bi-stable PUFs using IC aging phenomena. We described the use of NBTI in SRAM PUF for a limited improvement in reliability (40% reduction in errors). We then describe the use of HCI in modified SA PUF and present measured results that show that HCI-based DAA can generate response bits with a bit error rate $<5*10^{-9}$. This is equivalent to a 128-bit key failure rate $<10^{-6}$ which is typical targeted failure rate for error correction code (ECC) implementations.

Chapter 7

Conclusion and Future Work

7.1 Summary

In this work, we have discussed various aspects of the design of a Physical Unclonable Function (PUF). Based on measurement results, we provide an apples-to-apples comparison of security and VLSI metrics of two delay-based PUFs and two bi-stable based PUFs, including our proposed sense amplifier based PUFs. In the process of design and comparison, we have identified good design practices and have proposed several PUF design guidelines. We demonstrate that while randomness and uniqueness are relatively easy to achieve in most well designed PUF implementations, achieving high reliability of response is a challenge. Further, we find that the bi-stable PUFs are superior to delay-based PUFs in terms of VLSI metrics. We provide a brief overview of the overheads associated with conventional techniques of error correction (ECC) to achieve high reliability and then describe four techniques of reliability enhancement – multiple evaluation (ME), activation control (AC), directed accelerated aging (DAA), and post-silicon selection (PSS) – that aim to reduce the errors in the raw PUF response. To the best of our knowledge, we are the first to propose the use of aging phenomena to improve PUF reliability (DAA technique). We have also proposed the use of PSS in sense amplifier PUFs. Based on measured results, we demonstrate that both DAA and PSS can reduce PUF response errors to the extent that they can be achieve a 128-bit key error rate $< 10^{-6}$, the typical targeted failure rate of ECC implementations. We describe an autonomous and self-contained BIST controlled key generator that implements PSS on a sense amplifier PUF to generate a key with error rate $< 10^{-6}$ without using ECC. The key generator is fast, efficient, and unlike helper data generated in many ECC implementations, does not leak any information about the generated key bits. We describe the framework of a strong-PUF that is fast, reliable, and secure that uses the highly reliable bits from our key generator in conjunction with an AES cryptographic primitive to realize the strong-PUF.

7.2 Conclusions

Research on Physical Unclonable Functions (PUFs) is highly interdisciplinary by nature and researchers from the fields of circuit design theory, physics, mathematics, statistics, cryptography, and coding theory have made significant contributions. In this work, we address the challenges in the design of the PUF core, with a strong focus on circuit and implementation details.

PUFs are promising cryptographic primitives, but their use in commercial security applications is still limited, especially in the applications that require high reliability. We have presented several reliability enhancement techniques (by reducing errors in the raw PUF core) that are orthogonal to the conventional error correction techniques. With benefits over using the conventional error correction techniques alone, the use of error reduction techniques can result in more affordable designs, especially in resource limited systems like RFIDs.

7.3 Future Research Directions

The field of Physical Unclonable Functions is relatively new and most of the research in the field has happened in the last decade. A significant amount of work has been done to investigate various PUF core structures for the generation of random bits. However, proposals of new PUF core structures has slowed down over the last few years and it seems that simple and symmetric structures (e.g., SRAM, SA, ring oscillators) are able to generate bits with the desired properties. In principle, PUF core bits can be generated by leveraging uncertainties in a variety of IC manufacturing steps. However, for these bits to have good randomness properties, the PUF core should use phenomena that are well characterizable and are ideally perfectly random. Some of the most well researched and characterized PUF cores (e.g., SRAM, RO) derive randomness from device variations. These device variations, in turn, originate from phenomena like random dopant fluctuations and line edge roughness. These phenomena are heavily researched and well characterized by the IC process technology community and are generally considered to have nearly perfect random characteristics, making them the ideal phenomena to be leveraged for random bit generation. Overall, we feel that the design of PUF cores has been reasonably well understood and the main focus of research in the in the coming years would be in addressing issues that are preventing PUFs from large-scale use in security applications.

One of the main challenges is the provability of the randomness and reliability properties of PUFs. A good amount work has been done across research groups for randomness characterization and a more limited amount of work has been done for reliability characterization. However, most of the published work is based on small-scale emperical measurements or SPICE-like simulation results, and generally in slightly older technology nodes. A large-scale characterization of these properties in PUFs built in advanced technology nodes, and that is backed up by rigorous analytical modeling, will go a long way in making PUFs more trustworthy cryptographic primitives in security systems.

If PUFs are to be used in high volume manufacturing, it will be necessary to ensure that they are efficiently testable for their security metrics. The field of in-situ BIST-like testability of PUFs is yet largely unexplored and this needs to be addressed to make PUFs practically implementable.

Appendix A

Description of Testchips

A.1 Generation I Testchip

The Generation I testchip was designed in November 2010 with test structures of several PUF implementations to enable an apples-to-apples comparison. It was built using 65nm bulk CMOS technology from ST. The technology used had 7 metal layers, with top 2 used only for power routing. The technology provided two device processes: "Low-Power" or LP and "General-Purpose" or GP. Within each process, there were devices in three doping options: high-V_{TH} (hvt), low-V_{TH} (lvt), and standard-V_{TH} (svt). All the PUFs for comparison were built using LP-svt devices. Figure A.1 shows the dieshot of the testchip. On the 2mm^2 (1.4mmx1.4mm) testchip, we fabricated full custom implementations of arbiter, ring-oscillator, SRAM, and sense amplifier (SA) based PUFs, along with several unrelated projects. The details of the PUF types are summarized in Table A.1. The top-level layout snapshot of the PUF structures is shown in Figure A.2. The chips were packaged in a 132-pin PGA package and tested using a custom designed 4-layer printed circuit board (PCB) (Figure A.3). All PUFs were built using a symmetric design and layout to minimize any systematic variations (and hence maximize randomness and uniqueness) and by using near-minimum sized device sizes to maximize the spread of device characteristics (and hence maximize reliability), as explained in Section1.2.

A brief explanation of the structures on the testchip is provided in Section 3. A description of all the RO types built is shown in Table A.2. Relevant results from the various types of rings are discussed in Section 3.2.2.

Table A.1: Generation I Testchip

PUF Type	**Area (μm^2)**	**Device Type**	**Remarks**
Ring Oscillator	30k	all	256 rings
Arbiter	42k	LPsvt	12 64-bit chains
Sense amplifier	17k	LPsvt	Two 64x64 arrays (Latch-style + StrongARM)
SRAM	21k	LPsvt	Two 64x64 arrays

A.2 Generation II Testchip

The Generation II testchip was built in May 2012 that primarily contained novel circuits that implement reliability enhancement techniques and as well improved implementations of several test-structures from Generation I testchip. We used the same technology and manufacturer as in the Generation I testchip to be able to compare results across testchips and be able to re-use the already designed IPs (e.g., scan flops, buffers). Figure A.4 shows the dieshot of the testchip. The Generation II testchip is built in an area of $5.5mm^2$ (2.5mm x 2.2mm) that contained several non-related projects. The PUF test-structures are designed in the area of 2.5mm x 1mm and use 80 pads of the total 130 I/O pads. It contains test-structures for the HCI-based DAA (Section 6.3) and a key generator that implements PSS on SA PUFs (Section 5) that are built in an area of 0.32mm^2 and 0.125mm^2 respectively.

The details of the PUF types are summarized in Table A.3. The top-level layout snapshot of the PUF structures is shown in Figure A.5. The chips were packaged in a 132-pin PGA package and tested using a custom designed 4-layer printed circuit board (PCB) (Figure A.6).

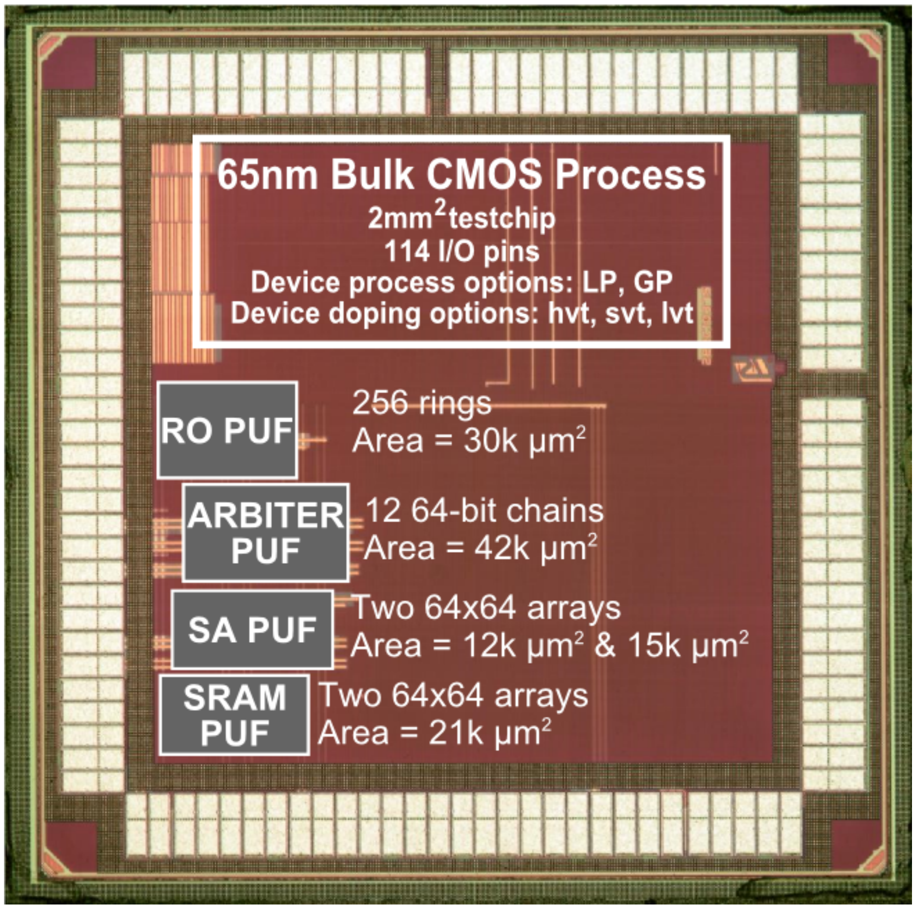


Figure A.1: Die micrograph of our Generation I PUF testchip. It is a 1.4mm x 1.4mm testchip built in 65nm bulk CMOS. The testchip has 114 I/O pins and contains PUF test structures that are built using LP and GP process options and various doping options (hvt, svt, lvt). The die includes other unrelated projects.

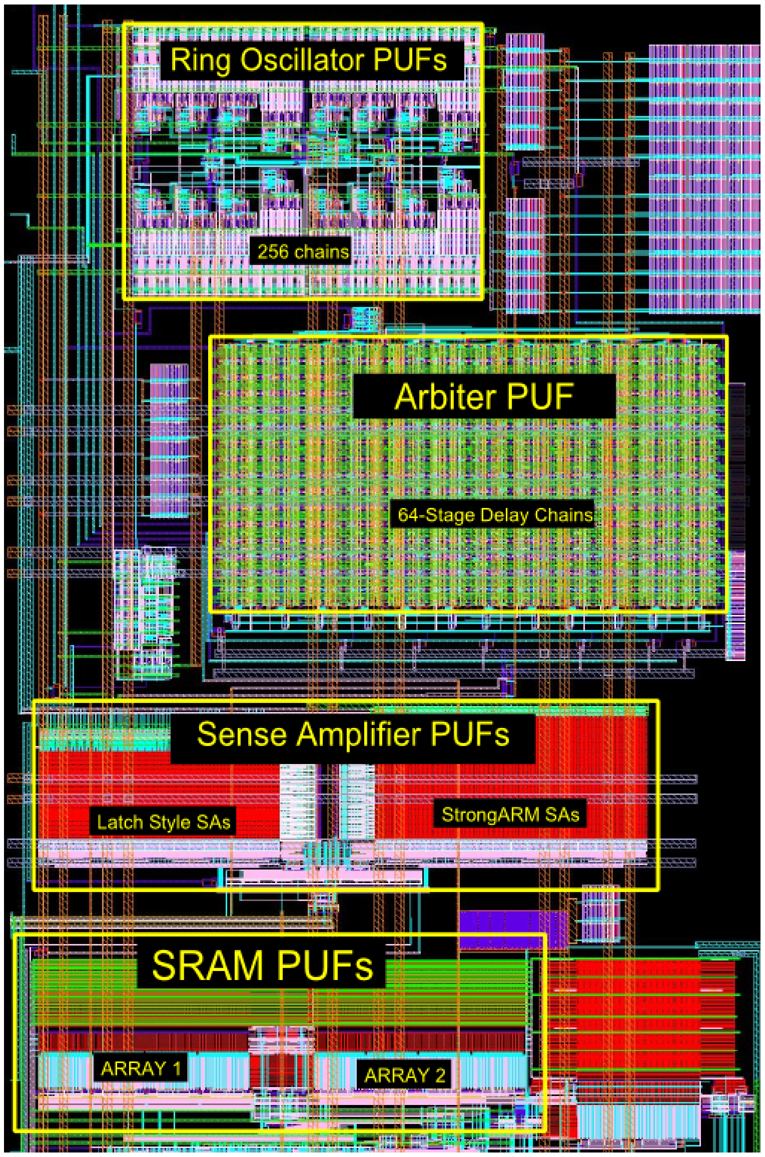


Figure A.2: Top-level layout snapshot of PUF structures on Generation I testchip

Table A.2: Description of 256 ROs on a die. Sizing$_1$ $\equiv$(W=150nm|L=60nm). Sizing$_2$ $\equiv$(W=280nm|L=120nm). Sizing$_3$ $\equiv$(W=750nm|L=60nm). All rings have a 2-input NAND with Sizing$_1$.

Design#	**#ROs/die**	**Device Doping**	**#stages/RO**	**Remarks**
1	32x2	SVT	13	Sizing$_1$
2	16x2	SVT	13	Sizing$_2$
3	16x2	SVT	13	Like #2,but 1 inv like #1
4	4x2	LVT	13	Like #1,but 1 tristated-inv
5	8x2	SVT	13	like #4
6	4x2	HVT	13	like #4
7	4x2	LVT	13	like #4, with tristate Sizing$_3$
8	8x2	SVT	13	like #7
9	4x2	HVT	13	like #7
10	16x2	SVT	17	like #1
11	16x2	SVT	31	like #1
TOTAL:	256	128 each built in LP and GP process devices		

Table A.3: Generation II Testchip

PUF Type	**Area (μm^2)**	**Device Type**	**Remarks**
HCI SAs	320k	GPsvt	3200 HCI-SAs
HCI test structures	220k	All	To study HCI-stress impact
Sense amplifier	259k	All	Six 64x64 arrays each (Latch-style + StrongARM)
SRAM	189k	All	Eight 64x64 arrays
Key Generator	125k	LPsvt	Implements PSS on SA PUFs

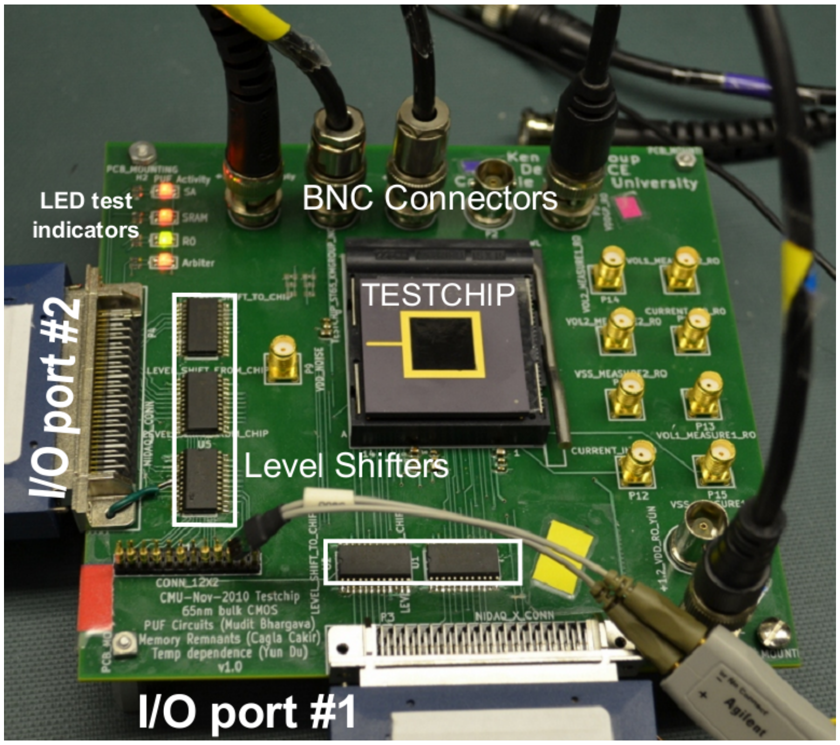


Figure A.3: Printed circuit board (PCB) to test Generation I Testchip

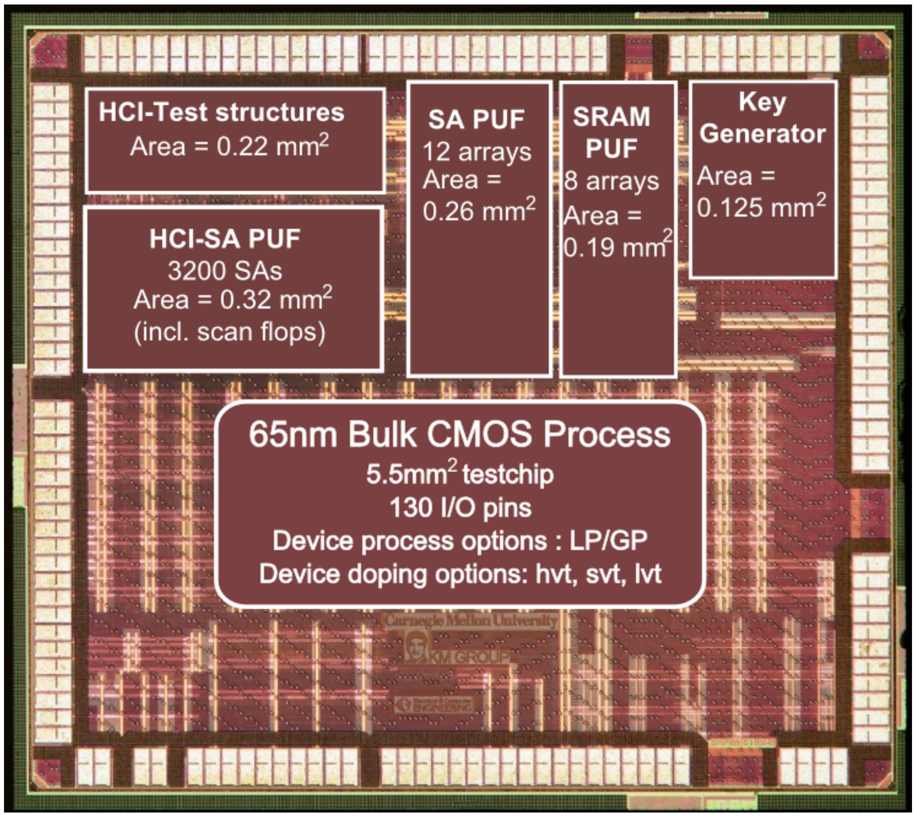


Figure A.4: Die micrograph of 2.5mm x 2.2mm, 130 I/O, Generation II testchip in 65nm bulk CMOS. There are 3200 HCI-SA PUF elements on each die in a total area of 0.32mm^2. The die included a number of other unrelated projects.

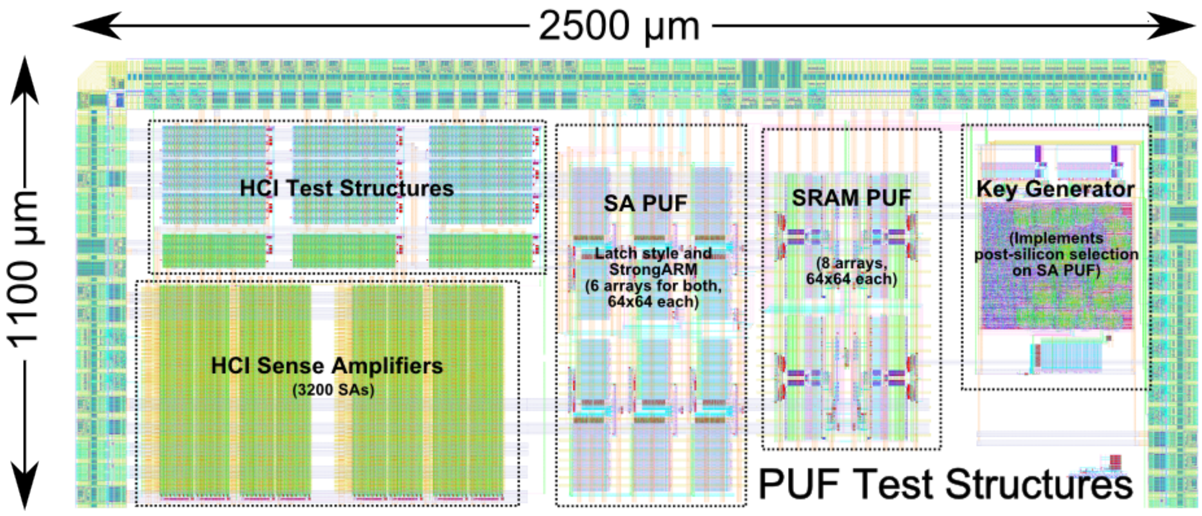


Figure A.5: Top-level layout snapshot of PUF structures on Generation II testchip

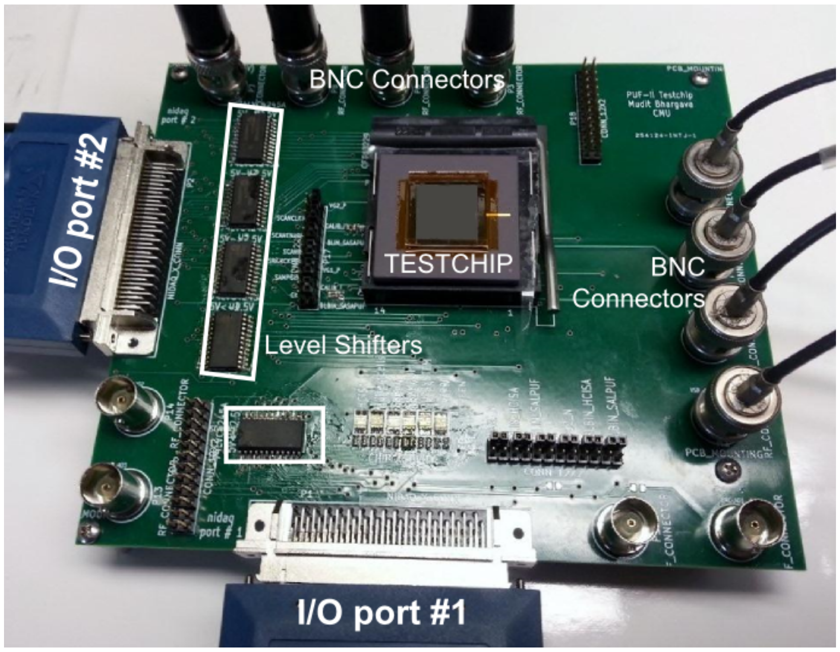


Figure A.6: Printed circuit board (PCB) to test Generation II Testchip

Appendix B

Sense Amplifier Offset Measurement Methodology

B.1 Sense Amplifier Array

For our measurements, we have implemented sense amplifiers (SA) in an array. These SAs are activated and accessed much like a SRAM. The inputs BLin and BLBin of each SA cell are tied together such the entire array has a single BLin and a BLBin signal. These are controlled directly from pads for the measurement of SA offsets.

B.2 Offset Measurement

To measure the offset, the input differential (i.e., the voltage difference between BLin and BLBin) is swept from -V_{OFFSET_MAX} to V_{OFFSET_MAX} in steps of V_{STEP}. At each step, all the SAs in the array are activated. This is done by activating one row of SAs at a time by sending a rising sense enable signal that is shared across SAs in a row (implemented like the wordline in a SRAM array). The activated SAs transfer their resolved values on BLout and BLBout, signals that are shared across SAs in a column (just like bitlines in a SRAM). In our design, BLout and BLBout generate full swing signals that are read out and post-processed to measure the offset of each SA in the array. For example, for a voltage sweep using $V_{STEP} = 10mV$, if the output of a particular

SA is “0” at $-30mV$ and “1” at $-20mV$, then we can say the SA has an offset between $-30mV$ and $-20mV$. Unless the measurements are repeated with a smaller V_{STEP}, we estimate the offset of the particular SA to be $\frac{1}{2} * \{(-30mV) + (-20mV)\} = -25mV$.

For our SAs, we chose $V_{OFFSET\_MAX} = 300mV$ and $V_{STEP} = 10mV$.

Appendix C

Test Infrastructure

C.1 Design

The test-structures were designed using the Cadence Virtuoso design platform and using 65nm bulk CMOS models and technology files from ST. The fabricated chips were packaged in a 132-pin PGA package and tested on a custom designed printed circuit board (PCB). The chips were fabricated through CMP. The packaging was done through Spectrum Semiconductor Materials, Inc. and the PCB was manufactured by Sierra Circuits, Inc. (Sierra Proto Express).

C.2 Testing

For testing, we used a National Instrument DAQ that acts as an interface between the test software (written in C) and the chip residing on the PCB. Agilent and Keithley power sources were used as power supplies. We used a TestEquity Model 107 temperature chamber to fully enclose the test PCB during temperature variation testing [3]. The generated data was processed using MATLAB.

Bibliography

[1] Intrinsic-id: Products webpage: www.intrinsic-id.com/products. 1.1

[2] NXP, Inc.: Press release (Feb 21, 2013): http://www.nxp.com/news/press-releases/2013/02/nxp-strengthens-smartmx2-security-chips-with-puf-anti-cloning-technology.html. 1.1

[3] Test Equity 107 Benchtop Temperature Chamber: http://www.testequity.com/products/1592. 1, C.2

[4] Verayo, Inc. Products webpage: www.verayo.com/products. 1.1

[5] K. Agarwal and S. Nassif. Characterizing Process Variation in Nanometer CMOS. In *Proceedings of 44th ACM/IEEE Design Automation Conference DAC '07*, pages 396–399, 2007. 2.3.2

[6] F. Ahmed and L. Milor. NBTI resistant SRAM design. In *Advances in Sensors and Interfaces (IWASI), 2011 4th IEEE International Workshop on*, pages 82 –87, June 2011. 6.2

[7] S. Bhardwaj, Wenping Wang, R. Vattikonda, Yu Cao, and S. Vrudhula. Predictive Modeling of the NBTI Effect for Reliable Design. In *Custom Integrated Circuits Conference, 2006. CICC '06. IEEE*, pages 189 –192, Sept. 2006. 6.4

[8] M. Bhargava, C. Cakir, and K. Mai. Attack resistant sense amplifier based PUFs (SA-PUF) with deterministic and controllable reliability of PUF responses. In *Proceedings of IEEE Int Hardware-Oriented Security and Trust (HOST) Symp*, 2010. 2.1, 2.3.2, 4.2.2, 5, 6.3

[9] M. Bhargava, C. Cakir, and K. Mai. Comparison of Bi-stable and Delay-based Physical Unclonable Functions from Measurements in 65nm bulk CMOS. In *Custom Integrated Circuits Conference, 2012. CICC '12. IEEE*, Sept 2012. 1.2.1, 1.2.2, 5, 5.1.3, 6.3

[10] M. Bhargava, C. Cakir, and K. Mai. Reliability enhancement of bi-stable PUFs in 65nm bulk CMOS. In *Hardware-Oriented Security and Trust (HOST), 2012 IEEE International Symposium on*, pages 25 –30, june 2012. 1.2.3, 5.3

[11] Christoph Bosch, Jorge Guajardo, Ahmad-Reza Sadeghi, Jamshid Shokrollahi, and Pim Tuyls. Efficient Helper Data Key Extractor on FPGAs. In Elisabeth Oswald and Pankaj Rohatgi, editors, *Cryptographic Hardware and Embedded Systems CHES 2008*, volume 5154 of *Lecture Notes in Computer Science*, pages 181–197. Springer Berlin / Heidelberg, 2008. 1.3, 3.7.1, 5, 6

[12] Heike Busch, Stefan Katzenbeisser, and Paul Baecher. PUF-Based Authentication Protocols Revisited. In Heung Youm and Moti Yung, editors, *Information Security Applications*, volume 5932 of *Lecture Notes in Computer Science*, pages 296–308. Springer Berlin / Heidelberg, 2009. 1.1

[13] S. Devadas, E. Suh, S. Paral, R. Sowell, T. Ziola, and V. Khandelwal. Design and Implementation of PUF-Based ”Unclonable” RFID ICs for Anti-Counterfeiting and Security Applications. In *Proceedings of IEEE Int RFID Conf*, pages 58–64, 2008. 1.1, 1.2.1, 1.2.2

[14] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and JanL. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 523–540. Springer Berlin Heidelberg, 2004. 1.3, 3.7.1

[15] S. Eiroa, J. Castro, M.C. Martinez-Rodriguez, E. Tena, P. Brox, and I. Baturone. Reducing bit flipping problems in SRAM physical unclonable functions for chip identification. In *Electronics, Circuits and Systems (ICECS), 2012 19th IEEE International Conference on*, pages 392–395, 2012. 5

[16] Blaise Gassend, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas. Silicon physical random functions. In *CCS '02: Proceedings of the 9th ACM conference on Computer and communications security*, pages 148–160, New York, NY, USA, 2002. ACM. 1.1, 1.2.1, 2.1, 2.3.1

[17] Blaise Gassend, Daihyun Lim, Dwaine Clarke, Marten van Dijk, and Srinivas Devadas.

Identification and authentication of integrated circuits. *Concurrency and Computation: Practice and Experience*, 16(11):1077–1098, 2004. 1.1, 1.2.1

[18] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls. Physical Unclonable Functions and Public-Key Crypto for FPGA IP Protection. In *Proceedings of Int. Conference Field Programmable Logic and Applications FPL 2007*, pages 189–195, 2007. 3.7.1, 5.2.3

[19] Jorge Guajardo, Boris kori, Pim Tuyls, Sandeep Kumar, Thijs Bel, Antoon Blom, and Geert-Jan Schrijen. Anti-counterfeiting, key distribution, and key storage in an ambient world via physical unclonable functions. *Information Systems Frontiers*, 11:19–41, 2009. 1.1

[20] Jorge Guajardo, Sandeep S. Kumar, Geert-Jan Schrijen, and Pim Tuyls. FPGA Intrinsic PUFs and Their Use for IP Protection. In *Proceedings of the 9th international workshop on Cryptographic Hardware and Embedded Systems*, CHES '07, pages 63–80, Berlin, Heidelberg, 2007. Springer-Verlag. 1.3, 2.4, 3.7.1, 5, 6

[21] H. Gustafson, E. Dawson, L. Nielsen, and W. Caelli. A computer package for measuring the strength of encryption algorithms. *Comput. Secur.*, 13(9):687–697, October 1994. 1.2.2, 3.4.2

[22] Peter Gutmann. Data remanence in semiconductor devices. In *Proceedings of the 10th conference on USENIX Security Symposium - Volume 10*, SSYM'01, pages 4–4, Berkeley, CA, USA, 2001. USENIX Association. 6

[23] Ghaith Hammouri, Erdin Ozturk, Berk Birand, and Berk Sunar. Unclonable lightweight authentication scheme. In Liqun Chen, Mark Ryan, and Guilin Wang, editors, *Information and Communications Security*, volume 5308 of *Lecture Notes in Computer Science*, pages 33–48. Springer Berlin / Heidelberg, 2008. 1.1

[24] A. Hastings. *The Art of Analog Layout*. Prentice Hall, Upper Saddle River, NJ, 2001. 1.2.2

[25] D. E. Holcomb, W. P. Burleson, and K. Fu. Power-Up SRAM State as an Identifying Fingerprint and Source of True Random Numbers. *IEEE Trans. Comput.*, 58(9):1198–1210, 2009. 1.1, 1.2.1, 1.2.2

[26] Daniel E. Holcomb, Wayne P. Burleson, and Kevin Fu. Initial SRAM state as a fingerprint

and source of true random numbers for RFID tags. In *In Proceedings of the Conference on RFID Security*, 2007. 1.2.1, 1.2.2, 2.1, 4.2.1

[27] Yongming Jin, Wei Xin, Huiping Sun, and Zhong Chen. PUF-Based RFID Authentication Protocol against Secret Key Leakage. In Quan Sheng, Guoren Wang, Christian Jensen, and Guandong Xu, editors, *Web Technologies and Applications*, volume 7235 of *Lecture Notes in Computer Science*, pages 318–329. Springer Berlin / Heidelberg, 2012. 1.1

[28] R. W. Keyes. Effect of randomness in the distribution of impurity ions on FET thresholds in integrated electronics. *IEEE J. Solid-State Circuits*, 10(4):245–247, 1975. 2.3.2

[29] Donald E. Knuth. The Art of Computer Programming, Volume II: Seminumerical Algorithms, 2nd Edition. 1981. 1.2.2, 3.4.2

[30] Unal Kocabas, Andreas Peter, Stefan Katzenbeisser, and Ahmad-Reza Sadeghi. Converse PUF-Based Authentication. In Stefan Katzenbeisser, Edgar Weippl, L. Camp, Melanie Volkamer, Mike Reiter, and Xinwen Zhang, editors, *Trust and Trustworthy Computing*, volume 7344 of *Lecture Notes in Computer Science*, pages 142–158. Springer Berlin / Heidelberg, 2012. 1.1

[31] Patrick Koeberl, Jiangtao Li, Roel Maes, Anand Rajan, Claire Vishik, and Marcin Wjcik. Evaluation of a PUF Device Authentication Scheme on a Discrete 0.13um SRAM. In Liqun Chen, Moti Yung, and Liehuang Zhu, editors, *Trusted Systems*, volume 7222 of *Lecture Notes in Computer Science*, pages 271–288. Springer Berlin / Heidelberg, 2012. 1.1, 1.2.1

[32] Farinaz Koushanfar. *Introduction to Hardware Security and Trust*, chapter Hardware Metering: A Survey, pages 103–122. Springer, 1st edition, 2012. 4.1.3

[33] S. S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls. Extended abstract: The butterfly PUF protecting IP on every FPGA. In *Proceedings of IEEE Int. Workshop Hardware-Oriented Security and Trust HOST 2008*, pages 67–70, 2008. 2.3.2

[34] Y. Lao and K.K. Parhi. Reconfigurable architectures for silicon physical unclonable functions. In *Electro/Information Technology (EIT), 2011 IEEE International Conference on*, pages 1–7, 2011. 1

[35] J. Lee, M. Tebranipoor, and J. Plusquellic. A low-cost solution for protecting IPs against

scan-based side-channel attacks. In *Proceedings of 24th IEEE VLSI Test Symp*, 2006. 6.2.4

[36] J. W. Lee, Daihyun Lim, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas. A technique to build a secret key in integrated circuits for identification and authentication applications. In *Proceedings of Digest of Technical Papers VLSI Circuits 2004 Symp*, pages 176–179, 2004. 1.1, 1, 2.4, 3.1.2

[37] Vincent Leest, Bart Preneel, and Erik Sluis. Soft Decision Error Correction for Compact Memory-Based PUFs Using a Single Enrollment. In Emmanuel Prouff and Patrick Schaumont, editors, *Cryptographic Hardware and Embedded Systems CHES 2012*, volume 7428 of *Lecture Notes in Computer Science*, pages 268–282. Springer Berlin Heidelberg, 2012. 1.3, 3.7.1, 5, 5.2.3, 6

[38] Daihyun Lim, J. W. Lee, B. Gassend, G. E. Suh, M. van Dijk, and S. Devadas. Extracting secret keys from integrated circuits. *IEEE Trans. VLSI Syst.*, 13(10):1200–1205, 2005. 1.1, 1, 2.4, 3.1.2

[39] R. Maes, V. Rozic, I. Verbauwhede, P. Koeberl, E. van der Sluis, and V. van der Leest. Experimental evaluation of Physically Unclonable Functions in 65 nm CMOS. In *ESSCIRC (ESSCIRC), 2012 Proceedings of the*, pages 486 –489, Sept. 2012. 1.2.1, 1.2.2, 6.3.5

[40] R. Maes, P. Tuyls, and I. Verbauwhede. A soft decision helper data algorithm for SRAM PUFs. In *Proceedings of IEEE Int. Symposium Information Theory ISIT 2009*, pages 2101–2105, 2009. 1.3, 3.7.1, 5, 6

[41] Roel Maes. *Physically Unclonable Functions: Constructions, Properties and Applications*. PhD thesis, Katholieke Universiteit Leuven, 2012. 2.4

[42] Roel Maes, Anthony Van Herrewege, and Ingrid Verbauwhede. PUFKY: A Fully Functional PUF-Based Cryptographic Key Generator. In *CHES*, pages 302–319. Springer, 2012. 3.7.1, 5.2.7

[43] Roel Maes, Pim Tuyls, and Ingrid Verbauwhede. Intrinsic PUFs from flip-flops on reconfigurable devices. In *Benelux Workshop on Information and System Security*, November 2008. 2.3.2

[44] Roel Maes, Pim Tuyls, and Ingrid Verbauwhede. Low-Overhead Implementation of a Soft

Decision Helper Data Algorithm for SRAM PUFs. In Christophe Clavier and Kris Gaj, editors, *Cryptographic Hardware and Embedded Systems - CHES 2009*, volume 5747 of *Lecture Notes in Computer Science*, pages 332–347. Springer Berlin Heidelberg, 2009. 1.3, 3.7.1, 5, 6

[45] Ken Mai. Introduction to hardware security and trust. SpringerLink : Bücher, chapter Side Channel Attacks and Countermeasures. Springer New York, 2012. 1.1

[46] Abhranil Maiti and Patrick Schaumont. Improved Ring Oscillator PUF: An FPGA-friendly Secure Primitive. *Journal of Cryptology*, 24:375–397, 2011. 10.1007/s00145-010-9088-4. 5

[47] M. Majzoobi, F. Koushanfar, and M. Potkonjak. Testing Techniques for Hardware Security. In *Proceedings of IEEE Int. Test Conference ITC 2008*, pages 1–10, 2008. 1, 2.4

[48] Mehrdad Majzoobi, Farinaz Koushanfar, and Miodrag Potkonjak. Techniques for Design and Implementation of Secure Reconfigurable PUFs. *ACM Trans. Reconfigurable Technol. Syst.*, 2(1):5:1–5:33, March 2009. 1

[49] George Marsaglia. The Marsaglia Random Number CDROM including the Diehard Battery of Tests of Randomness. 1.2.2, 3.4.2

[50] Amir Moradi, Axel Poschmann, San Ling, Christof Paar, and Huaxiong Wang. Pushing the limits: A very compact and a threshold implementation of aes. In KennethG. Paterson, editor, *Advances in Cryptology EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 69–88. Springer Berlin Heidelberg, 2011. 5.4.1

[51] P. Oldiges, Qimghuamg Lin, K. Petrillo, M. Sanchez, M. Ieong, and M. Hargrove. Modeling line edge roughness effects in sub 100 nanometer gate length devices. In *Proceedings of Int. Conference Simulation of Semiconductor Processes and Devices SISPAD 2000*, pages 131–134, 2000. 2.3.2

[52] Pappu. *Physical One way functions*. PhD thesis, 2001. 1.1.2

[53] Z. Paral and S Devadas. Reliable and efficient puf-based key generation using pattern matching. In *Hardware-Oriented Security and Trust (HOST), 2011 IEEE International Symposium on*, pages 128–133, 2011. 1.3, 3.7.1

[54] M.J.M. Pelgrom, A.C.J. Duinmaijer, and A.P.G. Welbers. Matching properties of MOS transistors. *IEEE J. Solid-State Circuits*, 24(5):1433–1439, October 1989. 1.2.2, 2.3.2

[55] G. Pobegen, T. Aichinger, M. Nelhiebel, and T. Grasser. Understanding temperature acceleration for NBTI. In *Electron Devices Meeting (IEDM), 2011 IEEE International*, pages 27.3.1 –27.3.4, Dec. 2011. 6.4

[56] Ulrich Ruhrmair, Heike Busch, and Stefan Katzenbeisser. Strong PUFs: Models, Constructions, and Security Proofs. In Ahmad-Reza Sadeghi and David Naccache, editors, *Towards Hardware-Intrinsic Security*, Information Security and Cryptography, pages 79–96. Springer Berlin Heidelberg, 2010. 2.4

[57] Ulrich Rührmair, Frank Sehnke, Jan Sölter, Gideon Dror, Srinivas Devadas, and Jürgen Schmidhuber. Modeling attacks on physical unclonable functions. In *Proceedings of the 17th ACM conference on Computer and communications security*, CCS '10, pages 237–249, New York, NY, USA, 2010. ACM. 1, 2.4, 3.1.2, 3.4.2, 3.5.1

[58] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangel, David Banks, Alan Heckert, James Dray, and San Vo. A statistical test suite for random and pseudorandom number generators for cryptographic applications. 2010. 1.2.2, 3.4.2

[59] G.-J. Schrijen and V. van der Leest. Comparative analysis of SRAM memories used as PUF primitives. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2012*, pages 1319–1324, 2012. 4.2.1

[60] Georgios Selimis, Mario Konijnenburg, Maryam Ashouei, Jos Huisken, Harmke de Groot, Vincent van der Leest, Geert-Jan Schrijen, Marten van Hulst, and Pim Tuyls. Evaluation of 90nm 6T-SRAM as Physical Unclonable Function for secure key generation in wireless sensor nodes. In *Proceedings of IEEE Int Circuits and Systems (ISCAS) Symp*, pages 567–570, 2011. 4.2.1, 4.2.2, 6

[61] P. Simons, E. van der Sluis, and V. van der Leest. Buskeeper PUFs, a promising alternative to D Flip-Flop PUFs. In *Hardware-Oriented Security and Trust (HOST), 2012 IEEE International Symposium on*, pages 7–12, 2012. 2.3.2

[62] R. Singh and N. Bhat. An offset compensation technique for latch type sense amplifiers in high-speed low-power SRAMs. *IEEE Trans. VLSI Syst.*, 12(6):652–657, 2004. 4.2.2

[63] B. Skoric, P. Tuyls, and W. Ophey. Robust Key Extraction from Physical Uncloneable Functions. In John Ioannidis, Angelos Keromytis, and Moti Yung, editors, *Applied Cryptography and Network Security*, volume 3531 of *Lecture Notes in Computer Science*, pages 407–422. Springer Berlin Heidelberg, 2005. 1.1

[64] Y. Su, J. Holleman, and B. Otis. A 1.6pJ/bit 96% Stable Chip-ID Generating Circuit using Process Variations. In *Solid-State Circuits Conference, 2007. ISSCC 2007. Digest of Technical Papers. IEEE International*, pages 406–611, 2007. 2.3.2

[65] G. E. Suh and S. Devadas. Physical Unclonable Functions for Device Authentication and Secret Key Generation. In *Proceedings of 44th ACM/IEEE Design Automation Conference DAC '07*, pages 9–14, 2007. 1.1, 1.2.1, 1.2.2, 2.1, 2.3.1, 5

[66] Pim Tuyls and Lejla Batina. RFID-Tags for Anti-counterfeiting. In David Pointcheval, editor, *Topics in Cryptology CT-RSA 2006*, volume 3860 of *Lecture Notes in Computer Science*, pages 115–131. Springer Berlin Heidelberg, 2006. 1.1

[67] Pim Tuyls and Boris Skoric. Secret Key Generation from Classical Physics: Physical Uncloneable Functions. In Satyen Mukherjee, RonaldM. Aarts, Raf Roovers, Frans Widdershoven, and Martin Ouwerkerk, editors, *AmIware Hardware Technology Drivers of Ambient Intelligence*, volume 5 of *Philips Research*, pages 421–447. Springer Netherlands, 2006. 1.1, 1.2.1, 1.2.2

[68] Vincent van der Leest, Geert-Jan Schrijen, Helena Handschuh, and Pim Tuyls. Hardware intrinsic security from D flip-flops. In *Proceedings of the fifth ACM workshop on Scalable trusted computing*, STC '10, pages 53–62, New York, NY, USA, 2010. ACM. 2.3.2

[69] V. Vivekraja and L. Nazhandali. Circuit-level techniques for reliable physically uncloneable functions. In *Hardware-Oriented Security and Trust, 2009. HOST '09. IEEE International Workshop on*, pages 30 –35, July 2009. 1.2.3, 4.2.1

[70] Li Wang, Qiuyi Ye, R. Wong, and M. Liehr. Product Burn-in Stress Impacts on SRAM Array Performance. In *Reliability physics symposium, 2007. proceedings. 45th annual.*

ieee international, pages 666 –667, April 2007. 6.1.1

[71] C.-E.D. Yin and Gang Qu. LISA: Maximizing RO PUF's secret extraction. In *Hardware-Oriented Security and Trust (HOST), 2010 IEEE International Symposium on*, pages 100–105, 2010. 3.7.1, 5

[72] H. Yu, P.H.-W. Leong, H. Hinkelmann, L. Moller, M. Glesner, and P. Zipf. Towards a unique FPGA-based identification circuit using process variations. In *Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on*, pages 397–402, 2009. 5

[73] Meng-Day Yu and S. Devadas. Secure and Robust Error Correction for Physical Unclonable Functions. *IEEE Design & Test of Computers*, 27(1):48–65, 2010. 1.2.3, 1.3, 3.7.1

[74] Meng-Day Mandel Yu, David M'Raihi, Richard Sowell, and Srinivas Devadas. Lightweight and secure PUF key storage using limits of machine learning. In *Proceedings of the 13th international conference on Cryptographic hardware and embedded systems*, CHES'11, pages 358–373, Berlin, Heidelberg, 2011. Springer-Verlag. 1.3, 3.7.1