Detection of Fraudulently Recycled Integrated Circuits

by

Alexandros Dimopoulos

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in the Department of Electrical & Computer Engineering

University of Victoria

We acknowledge and respect the Lək̓ʷəŋən (Songhees and Xʷsepsəm/Esquimalt) Peoples on whose territory the university stands, and the Lək̓ʷəŋən and W̱SÁNEĆ Peoples whose historical relationships with the land continue to this day.

Detection of Fraudulently Recycled Integrated Circuits

by

Alexandros Dimopoulos

Supervisory Committee

Dr. Mihai Sima, Supervisor

Department of Electrical & Computer Engineering

Dr. Stephen Neville, Co-Supervisor

Department of Electrical & Computer Engineering

Dr. Sudhakar Ganti, Outside Member

Department of Computer Science

Abstract

As is true of many types of products, integrated circuits (ICs) are subject to counterfeiting. Of the various methods of counterfeiting ICs, recycling, where still functioning units are recovered from waste streams and passed off as new, is particularly problematic. These components are still functional but are unreliable due to their unknown operational histories. This poses an important risk to critical infrastructure in domains ranging from defense, to health care, to transportation. The long lifetimes of these systems makes it challenging to find suitable replacement ICs since the electronics industry evolves rapidly and produces parts with comparatively short production lives. A number of recycled integrated circuit (IC) risk mitigation approaches have been proposed, but these generally lack pragmatic feasibility. This work proposes a novel real-world deployable on-chip sensor that: 1) is tamper-resistant by exploiting persistent changes caused by hot carrier injection (HCI); 2) generates a DC signal measurable by common low-cost test equipment; and 3) reuses an existing I/O interface, including existing pins; while 4) requiring a very small footprint. Combining this sensor with a random sample-based testing strategy allows for low-cost and time efficient detection of fraudulently recycled batches of ICs. Employing a random sample size as small as 130 is sufficient for identifying such batches with a statistical significance level of 0.01. This is demonstrated through simulation-based validation using process-accurate models of a 65 nm technology.

The design of countermeasures against IC recycling requires the ability to simulate aging in CMOS devices. Electronic design automation tools commonly provide this ability; how-

ever, their models must be tuned for use with a specific target technology. This requires data which is ideally provided by a fab. It may also be collected from a set of purpose-built test devices, a costly and time-consuming process. This work describes a novel, low-cost, and rapid approach to tuning such models. It is an iterative method that leverages public domain data sourced from published studies to fit an aging model. Results are statistically validated against the target technology's specification. The approach is demonstrated by fitting a compact hot carrier injection degradation model for use with both core and I/O nMOSFETs from a specific 65 nm technology. Resulting model parameter values are validated with a maximum error of 0.5 % with a 99 % confidence bound.

Table of Contents

Supervisory Committee **ii**

Abstract **iii**

Table of Contents **v**

List of Figures **viii**

List of Tables **x**

1 Introduction **1**

2 Literature Review **10**

2.1 Aging in Integrated Circuits 12

2.1.1 Bias Temperature Instability 12

2.1.2 Hot Carrier Injection 14

2.1.3 Time Dependent Dielectric Breakdown 15

2.1.4 Electromigration 16

2.2 Recycling Countermeasure Prior Art 17

2.2.1 Preemptive Methods 18

2.2.2 Passive Methods 28

3 Initial Design Study of a HCI-based IC Anti-Recycling Sensor **41**

3.1 Design Considerations 43
3.1.1 Aging Mechanism 44
3.1.2 Sensor Signalling 45
3.1.3 Measurement Interface 46
3.1.4 Measurement Accuracy 47
3.2 Initial Performance Estimates 48

4 Circuit Simulation and Data Analysis Toolchain 56
4.1 Toolchain Description 56
4.2 Toolchain Use 60
4.2.1 TSMC SPICE Models 60
4.2.2 MOSRA 61

5 Leveraging Public Information to Fit a Compact Hot Carrier Injection Model to a Target Technology 65
5.1 Model Parameter Search Process 66
5.1.1 General Description 67
5.1.2 Application Example 70
5.2 Data Sources 75
5.3 Model Fitting Results 80
5.4 Circuit Level Validation 83

6 A Small Tamper-Resistant Anti-Recycling IC Sensor With a Reusable I/O Interface and DC Signalling 87
6.1 Sensor Design 89
6.2 Sensor Operation 92
6.3 Results and Discussion 93

7 Conclusions 103

7.1 Summary of Work 105
7.2 Future Work 108

Bibliography **112**

A hci_param_tune.m **122**

B hci_core_mc.sp **127**

C radeg_pull.py **129**

D mc_fail_stats.py **134**

E mc_readmode.sh **137**

F tharcdc.sp **140**

G hspice_RAN_pull.py **145**

H hspice_collate_MC_RAN.py **152**

I tharcdc_waldtest.m **154**

List of Figures

1.1 Taxonomy of IC Counterfeiting. 3

2.1 The bathtub curve. 11
2.2 BTI causes a temporary shift in V_{TH}. 13
2.3 TDDB is difficult to localize in time. 16
2.4 A pair of ring oscillators tracks BTI degradation. 18
2.5 The N-CDIR sensor. 20
2.6 A subthreshold RO stage. 22
2.7 An aging sensor which exploits electromigration [45]. 25
2.8 Histogram of ages when EM sensor output flips from 0 to 1 [44]. 25
2.9 Ring oscillator configuration on an FPGA. 32
2.10 I_{DDT} difference between a stressed and unstressed multiplier. 39

3.1 Proposed reuse of an I/O for a sensor interface. 46
3.2 Bayes error definition. 54
3.3 Bayes error when detecting an HCI effect from a differential threshold voltage measurement. 55

4.1 MOSRA simulation flow consists of two simulations. 57

5.1 Degradation model parameter search process. 68
5.2 Implementation of the fitting method for the MOSRA level 3 HCI model. . . 71

5.3 Determining an appropriate weight for Λ. 74
5.4 HCI measurements from [71] 77
5.5 HCI measurements from Yuan et al. [72]. 77
5.6 HCI measurements from [67]. 78
5.7 HCI measurements from Xie et al. [73] and Ren et al. [74]. 79
5.8 Method for testing the accuracy bounds of our HCI model. 85
5.9 Inverter degradation according to MOSRA with $\mathbf{v_{core}}$. V_{DD} is 1.5 V. Also shown are the D_{H} points, each mapped onto its corresponding curve. 86

6.1 THARC-DC schematic. 90
6.2 Simulation and analysis workflow for THARC-DC. 94
6.3 $\mathrm{M_{age}}$ degradation in **AGE** mode. 95
6.4 Average static current in THARC-DC over time. Operating conditions are those defined in Table 6.1. 95
6.5 $\mathrm{I_{IN}}$ under **READ** mode. 96
6.6 Illustration of the Wald test. 98
6.7 Minimum identifiable batch age as a function of sample size and operating condition of the batch. 99
6.8 THARC-DC authentication process. 101

List of Tables

2.1 Process variations used in simulations of CDIR sensors [41]. 21

2.2 Performance simulations of CDIR sensors [39], [40], [41] 21

2.3 Detection accuracy through path delay. 30

3.1 HCI model parameters for an IMEC 65 nm process. 50

3.2 65 nm TSMC device parameters 51

3.3 V_{TH} difference due to HCI and due to process variations. 52

3.4 Mean HCI-induced V_{TH} shift expressed in terms of intrinsic V_{TH} mismatch. . 52

5.1 MOSRA HCI Compact Model Variables [62]. 71

5.2 MOSRA HCI Compact Model Parameters [62]. 72

5.3 Public domain data used for determining the parameter sets. 80

5.4 Derived MOSRA HCI parameter values. 81

5.5 MOSRA model parameter search process settings. 81

5.6 MOSRA model parameter validation. All values have units of years. 82

5.7 RMS relative error for data set simulations using the MOSRA model and TSMC 65 nm device models. 82

5.8 Inverter stimuli [67]. 83

5.9 Reported inverter nMOSFET $I_{\mathrm{D,sat}}$ percentage degradation from measurement (D_{H}) and simulation (S_{H}) [67]. 83

6.1 Expected $\mathrm{V_{G,age}}$ and $\mathrm{V_{D,age}}$ values under different operating conditions. . . . 92

Chapter 1

Introduction

THE global trade in counterfeit goods is a serious and ever increasing problem. The Organization for Economic Co-operation and Development (OECD) and the EU Intellectual Property Office estimate that this illicit trade accounted for 2.5 % of global trade in 2019 [1]. The economic impact is disproportionately felt in developed countries whose economies are increasingly reliant on intellectual property (IP). Fake goods also pose a safety risk due to their usually inferior quality [1]. For example, in a test of 400 fake iPhone adapters, UL Canada found that 397 failed basic safety tests with 12 posing a lethal risk of electrocution. Fake components have been found in many places including Canadian military aircraft and Air Force One [2].

An increasingly consequential sector of counterfeits is integrated circuits (ICs). The Semiconductor Industry Association has estimated that in 2017 these counterfeits cost the chip industry $7.5 billion per year [3]. This figure is likely to have increased since due to the severe supply chain disruptions during the COVID pandemic [3]. Beyond the economic toll, counterfeit electronic components can severely impact the safety and reliability of critical systems in areas such as health care, transportation, and defense [4]. Due to the explosion in consumer electronics since the 1980s, it is no longer economical to have specialized production and distribution chains for these applications. This transformation is starkly apparent in

the case of military applications. In the 1980s military applications accounted for 26 % of all semiconductor demand, today it is less than 0.1 %. This means that for cost reasons, military and other critical systems now depend upon supply chains that are completely intertwined with those for consumer products [5].Very few vertically integrated semiconductor companies exist today. The industry is now mostly horizontally integrated and globally distributed. Supply chains are of such high complexity that they lie outside the observability and control of individual companies and countries. The resulting loss of trust in supply chains has created openings for counterfeiters to operate amongst the many layers of independent parts brokers and distributors, both domestic and foreign, responsible for moving many ICs from original equipment manufacturers (OEMs) to customers[6]. In US government investigations, almost all of the discovered counterfeits were tracked back to such distributors [5]. Figure 1.1 shows a simplified representation of the different categories of counterfeit ICs entering the supply chain. These categories are:

- **Remarked and Out of Specification:** Manufacturing process variations lead to performance variations in produced ICs, which are sorted and graded for different applications. For example, ICs which continue to perform under a wide temperature range may be marked for industrial or military use, leaving those that are more temperature sensitive destined for consumer applications. The industrial and military grade components command a higher price, making it tempting for a counterfeiter to remark legitimate consumer grade parts as military grade. The remarked parts may also include those that have failed minimum performance requirements and should have been disposed of [6].

- **Overproduced:** In this scenario, an untrusted pure play fab produces more parts than it was contracted for. It then sells its extra stock, with the profit entirely bypassing the legitimate IP owner [6]. These counterfeits may be as reliable as legitimate components.

- **Cloned and Fake:** A pirate fab that has no relationship with an IP's owner may produce cloned ICs after acquiring a design through theft or reverse engineering [6]. In the extreme case, such a fab will produce any component requested by a customer, including those that do not exist. A US government investigation identified such fabs in China and was successfully able to order ICs with part numbers that do not correspond to any existing design. These are large operations (10 000 - 15 000 employees) which operate in the open [5]. What exactly is in such ICs is a good question.

- **Recycled:** Electronic waste (e-waste), consisting of broken and obsolete electronics, is commonly collected for deconstruction so that precious materials inside can be recovered and recycled. Much e-waste is comprised of functional but obsolete products, and even broken products will contain some still functional ICs. In this scheme, e-waste is disassembled and ICs are removed from PCBs. Still functioning ICs are refurbished, remarked, and sold as new [5], [6].

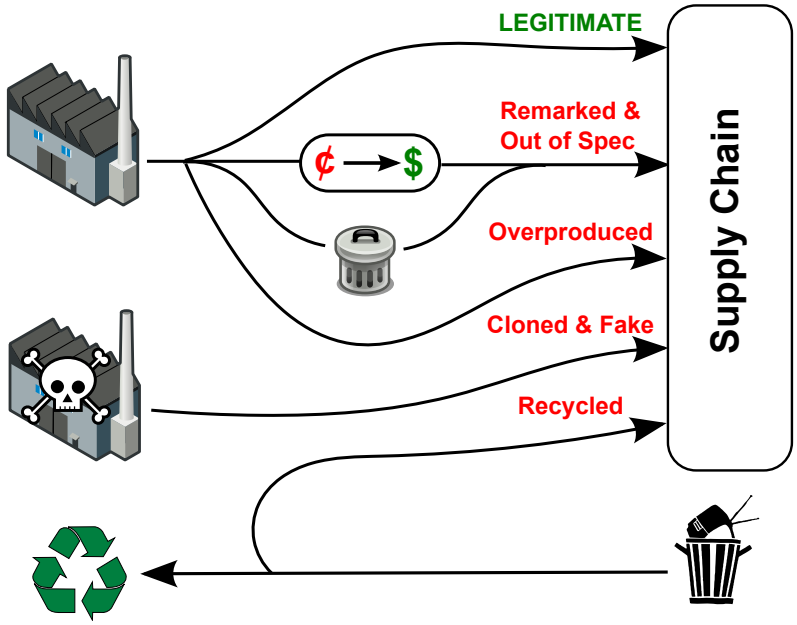


Figure 1.1: In a globally distributed production environment there are many opportunities for illegitimate ICs to enter supply chains.

Unless the global semiconductor industry dramatically changes its production methods, the problem of counterfeit ICs will only get worse. Combatting counterfeiters entails either preventing them from practising their craft or identifying their handiwork before it is used. A universal solution to this multifaceted problem is unlikely. For example, any detection scheme for finding counterfeits present in the supply chain are unlikely to identify overproduced ICs since these are indistinguishable from legitimate components. Rather, a divide and conquer approach where each type of counterfeit is individually attacked will likely be needed.

Recycled components are a particularly vexing class of counterfeits which have been identified by the U.S. Department of Defense as the greatest threat to the reliability of its systems [5]. CMOS ICs are all affected by aging mechanisms which degrade their performance with use [7]. Eventually this degradation is significant enough that an aged circuit can no longer operate within specifications. Recycled components are still functional; however, their true ages and remaining useful lifetimes are unknown, leading to unpredictable system failures, increased maintenance costs, and potential catastrophes. This is poised to become a much larger problem as the amount of generated e-waste continues to grow. An estimated 50 million tonnes of e-waste will be produced around the world this year. A combination of rising global incomes, lowering cost of electronics, and planned product obsolescence make e-waste the fastest growing waste stream globally [8]. The enormous and growing stock combined with increasing demand for ICs will only make counterfeiting by recycling more tempting. A likely and consequential new destination for these devices will be the internet of things (IoT). For several years now, the number of internet-connected devices has surpassed the number of humans. The global count of IoT devices was estimated to be 18.8 billion at the end of 2024 [9] and is estimated to grow to 38.8 billion by 2029 [10], thus representing the fastest growing economic sector in human history [11]. These devices have applications in important systems such as traffic control, medical aids, and industrial automation [12]. The economics of IoT means that profit margins on individual devices are likely to be razor thin, leaving very little motivation for extensive testing. This is a problem because IoT devices

will be deeply embedded in the environment, likely making repairs difficult and expensive. Finding the recycled ICs which will undoubtedly end up in the IoT will be challenging. Detection methods will have to be inexpensive and easy to use.

Combatting recycled ICs is challenging due to the asymmetry of the problem: counterfeits are easily produced and difficult to identify. Currently, the only accepted way of detecting recycled ICs is through physical inspection [6]. The pins and package can be inspected under a microscope to look for signs of wear, old solder, and rubbed out markings. The IC's internal structure can also be examined for signs of material defects. Some of the inspection methods are destructive and all are extremely time consuming. Clear standard definitions of counterfeits is also lacking so determination ultimately falls to a human expert's judgement. Physical inspection cannot be scaled up to properly deal with the magnitude of this problem.

Because recycled ICs have undergone some degradation due to usage, combatting the problem first requires an understanding of the major degradation mechanisms affecting CMOS: bias temperature instability (BTI), hot carrier injection (HCI), time dependent dielectric breakdown (TDDB), and electromigration. These are reviewed in chapter 2. A number of non-destructive countermeasures that depend upon electrical measurements have been proposed, many of which are reviewed in chapter 2. They fall into two broad categories: preemptive methods, which aim to ease counterfeit detection by modifying an IC design before fabrication, typically by including a specialized sensor; and passive methods, which seek to identify counterfeits purely through non-destructive measurements performed on ICs. None of the proposed approaches fully satisfy the requirements for a viable and scalable countermeasure:

- **Easy to deploy:** A counterfeit detection scheme cannot be successful without the support of IC designers. It should aim to be cheap and easy to deploy. It should not compromise a designer's IP by for example, requiring publication of sensitive data, or facilitating reverse engineering. Any on-chip sensors should minimize their area and performance overheads. They should be small and avoid interacting with the host

system. Additional I/O pins should also be avoided. Finally, they should be compatible with a standard CMOS process.

- **Easy to use:** Different users ranging from IC brokers to board-level integrators may be interested in testing for recycled components. These users may also vary greatly in size from small startups up to global multinationals, and have access to an equivalently wide disparity in resources. A detection scheme should therefore be inexpensive and easy to use for all of them. A test should be fast to perform and require minimally complex equipment.

- **Provide a clear result** An IC's legitimacy, or illegitimacy, should be obvious. A low error rate will require robustness against manufacturing process variations. The measurement resolution should not be overwhelmed by the process variation.

- **Tamper-resistant** An easily circumvented security measure provides no security. Counterfeiters should not be able to defeat the detection scheme without incurring a significant cost.

This dissertation proposes a novel countermeasure which meets the above design criteria. In chapter 3, the case is made that this countermeasure should be a sensor because it can be designed to generate a clear and easily measured signal. This sensor should have the following features:

- It should exploit the hot carrier injection (HCI) degradation mechanism. Both TDDB and electromigration have been proposed as a basis for an embedded anti-recycling sensor. The permanence of these mechanisms is attractive, however there are downsides. Both are stochastic in nature, making it difficult to control the timing of the failures they produce. There are also practical limitations in terms of area or required voltages. Sensors which exploit BTI have also been proposed. This is a well-studied degradation mechanism, which is considered a major issue in IC reliability. However,

BTI largely reverses once the responsible stress voltage is removed. A recycled IC may spend significant time unpowered between when it is discarded and resurrected as a counterfeit. HCI, on the other hand does not reverse once the responsible stress is removed.

- It should produce a DC signal so that measurements can be made quickly and by using only simple equipment. Because a DC signal allows for speedy and economical testing, such a sensor could allow even small startups to validate their supplies.

- It should piggy-back on top of an existing I/O interface to eliminate the need for additional pins. Because the signal is DC, this can be done quite simply and in a way that will not interfere with the normal operation of an IC.

The second half of chapter 3 presents preliminary performance estimates of an HCI-based sensor. These suggest that within six months such a sensor is capable of successfully differentiating HCI aging from process variations.

Chapter 4 discusses the different tools employed in this work as well as their use. Design and performance assessments were carried out using Synopsys HSPICE, a commercial SPICE simulator, which includes facilities for simulating the effects of BTI and HCI on circuits. SPICE device models for a real 65 nm technology were used in this work in order to achieve realistic results.

When considering recycled ICs, it is crucial that the effects of CMOS degradation mechanisms on circuit performance be accurately predicted. In order to predict system failures, all major EDA tools are equipped with facilities to simulate these effects mechanisms. The degradation models are not generic and must be tuned to a target CMOS technology if they are to produce accurate results. Fabs are the authoritative source of degradation models and the data required to tune EDA models to their technologies, but this is sensitive information which they tightly control and do not always share with academic researchers. In such sit-

uations, it is possible to use the novel fitting method presented in chapter 5, which exploits public domain data.

Unlike proposed countermeasures, which have sought to authenticate individual ICs through accurate and precise determination of their ages, this work focuses on batch authentication. This different approach has two advantages. First, random sampling methods would be used in any realistic scenario since exhaustive testing would be enormously expensive at large scales. Second, recycled ICs are likely to range in age on the scale of months to years so only a rough age estimate suffices for authentication. Through statistical means, such a sensor's signal can allow accurate determination of a batch's authenticity even if this is not possible at the level of an individual IC. Because it does not need to make highly detailed age measurements, its design can be simple as can be the information it communicates. Proposed designs have complicated designs and interfaces, including features such as memories, state machines, frequency counters, and even dedicated pins. By designing the sensor to generate a DC signal, measurements can be made quickly and by using only simple equipment. Because a DC signal allows for speedy and economical testing, such a sensor could allow even small startups to validate their supplies. Chapter 6 presents a small anti-recycling sensor designed for the authentication of entire batches of ICs.

The major contributions presented in this dissertation are:

- A general method for fitting a MOSFET degradation model to a target technology by leveraging public domain data. The solution produces results which are statistically validated against the target technology's specifications. This contribution was published in the IEEE Access Journal [13].

- A novel compact IC sensor for the identification of IC batches containing recycled counterfeits. This contribution was published in the IEEE Open Journal of Circuits and Systems [14] and an initial feasibility study of the idea was presented at the IEEE

61st International Midwest Symposium on Circuits and Systems [15]. The sensor has the following features:

1. has improved tamper-resistance through exploiting hot carrier injection (HCI), the effects of which do not reverse once the responsible stress is removed;
2. produces an DC signal that is easily measurable via standard low-cost testing equipment;
3. makes use of existing I/O interfaces, thereby requiring no specialized interface circuitry, nor any extra pins;
4. has a comparatively small footprint, allowing easy integration into an existing IC design;
5. is designed to allow for low-cost, coarse time resolution age detection suitable to efficiently assess fraudulent reuse within larger volume IC batches.

The rest of this dissertation is organized as follows. Chapter 2 provides required background information on CMOS degradation mechanisms and a literature review on proposed methods for combatting IC recycling. None of the proposed countermeasures reviewed provide a satisfying solution to the problem of IC recycling. Chapter 3 presents an initial design study of a new approach, a hot-carrier injection (HCI) based anti-recycling sensor. The work presented in this dissertation was simulation based. Chapter 4 discusses the toolchain that was used. Predicting the performance of an HCI based sensor required the ability to accurately simulate HCI-induced degradation in MOSFETs. Chapter 5 discusses fitting an industry-standard HCI model for the simulation of HCI-induced degradation in a process-accurate MOSFET model. Chapter 6 presents the design and simulation of a novel anti-recycling sensor which exploits HCI-induced degradation. Finally, Chapter 7 gives conclusions and discusses possible future directions.

Chapter 2

Literature Review

AS with all engineered systems, ICs experience stresses while operating, leading to shifts in their performance parameters and eventually to critical failures. In reliability engineering, a widely used model to describe the pattern of failures in time is the so-called "bathtub curve" shown in figure 2.1. The curve describes the time evolution of a system's hazard rate, the conditional probability of a failure occurring on the interval $(t, t + dt)$ given that no failure has occurred by time t. The curve can be split into three sections: an initial period of rapidly decreasing hazard rate called *infant mortality*, a period where the hazard rate is small and rather stable called the *operating life*, and a period of rapidly increasing hazard called the *wearout* [16]. Ideally, a system should not reach wearout before the end of its design life.

Failures during infant mortality are due to manufacturing defects that remain undiscovered during initial testing. ICs are subjected to burn-in testing before being delivered to customers in order to uncover these defects. During this short period of high-stress operation, involving abnormally high voltages and temperatures, defective ICs fail and are weeded out. The remaining ICs are at the beginning of their operating life stage, where the hazard is lowest.

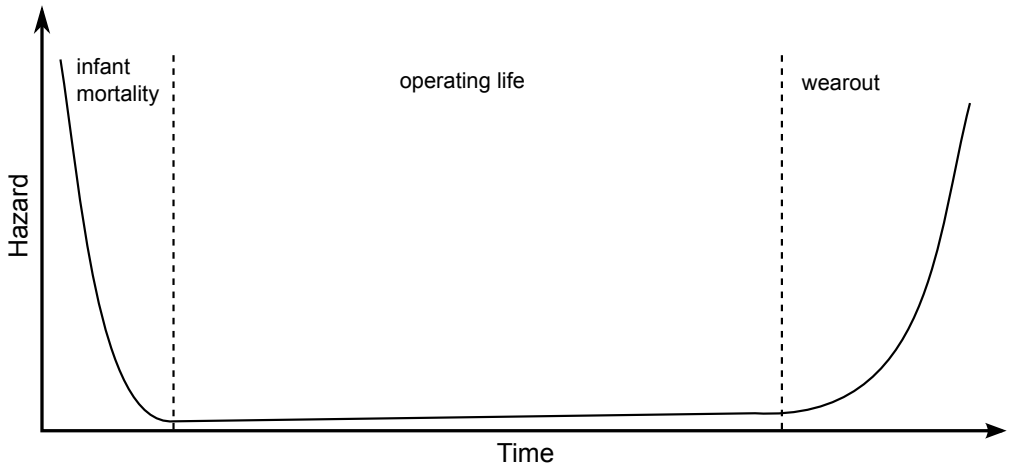


Figure 2.1: The instantaneous failure rate, or hazard, over the lifetime of an integrated circuit resembles a bathtub.

Once an IC is deployed it should operate with very low failure rates over its operating lifetime before transitioning into its wearout stage at the end of its life. A number of factors determine the time to wearout such as the technology, manufacturing processes, and operating conditions such as temperature, supply voltage, and duty cycle. [17].

Before any IC is released, it will undergo extensive reliability analysis in order to accurately predict its operating lifetime. As a result, much effort has been, and continues to be, spent on understanding the failure mechanisms which affect CMOS. Recycled ICs pose a threat because their true remaining operating lifetimes are unknown, leading to unpredictable early system failures. Because recycled ICs have not yet failed, identifying them requires knowledge of the failure mechanisms and their effects on devices and circuits.

The rest of this chapter consists of an overview of the important failure mechanisms currently affecting CMOS (section 2.1) and a review of proposed strategies to exploit them for the detection of recycled ICs (section 2.2).

2.1 Aging in Integrated Circuits

CMOS is affected by four major failure mechanisms: bias temperature instability (section 2.1.1), hot carrier injection (section 2.1.2), time dependence dielectric breakdown (section 2.1.3), and electromigration (section 2.1.4).

2.1.1 Bias Temperature Instability

Bias temperature instability (BTI) causes a MOSFET's threshold voltage (V_{TH}) to gradually increase in an absolute value sense over time when the MOSFET channel is inverted ($|V_{\mathrm{GS}}| > |V_{\mathrm{TH}}|$) [17]. The mechanism is referred to as negative temperature bias instability (NBTI) in pMOSFETs and positive temperature bias instability (PBTI) in nMOSFETs. The distinction refers to the polarity of V_{GS} needed to invert each device's channel [18].

BTI is due to charge accumulation at the channel-oxide interface, though the exact physical mechanism is not well understood. One model proposes the generation of interface states when Si-H bonds at the interface are broken. Another model posits that the accumulated charge is captured by defects in the gate dielectric [17], [19].

Though NBTI has been known for some 50 years, it has only relatively recently come to widespread attention[18]. It is now considered to be one of, if not the most, important issue in CMOS reliability [17], [18], [20], [21], [22], [23], [24], [25], [26]. Technological changes introduced to allow scaling beyond 130 nm have created favourable conditions for NBTI degradation during regular circuit operation. For example, the reduction in operating voltage is not keeping pace with CMOS dimension scaling, leading to stronger gate-channel electric field strengths [17]. This is due to the need to maintain a large difference between a MOSFET's on (I_{ON}) and off (I_{OFF}) currents in the face of a constant sub-threshold slope [27]. The introduction of high-k dielectric materials in the gate stack has caused PBTI to now also become a reliability concern [18].

The shift in threshold voltage (ΔV_{TH}) due to BTI has a power law dependence on both the applied gate-to-source voltage (V_{GS}) and the time the MOSFET is stressed (t) [17]. There is also a dependence on the electrically determined oxide thickness (t_{ox}), the temperature (T), and trap activation energy (E_a), as shown in (2.1) [17], [18], [21]. The exponent for the stress time dependence usually ranges from about 0.1 to 0.2 [19]. This power law in time holds over many decades but is not constant, it eventually saturates [17].

$$\Delta V_{\mathrm{TH}} \propto \left(\frac{V_{\mathrm{GS}}}{t_{ox}}\right)^{m} \cdot exp\left(\frac{-E_a}{kT}\right) \cdot t^{n} \tag{2.1}$$

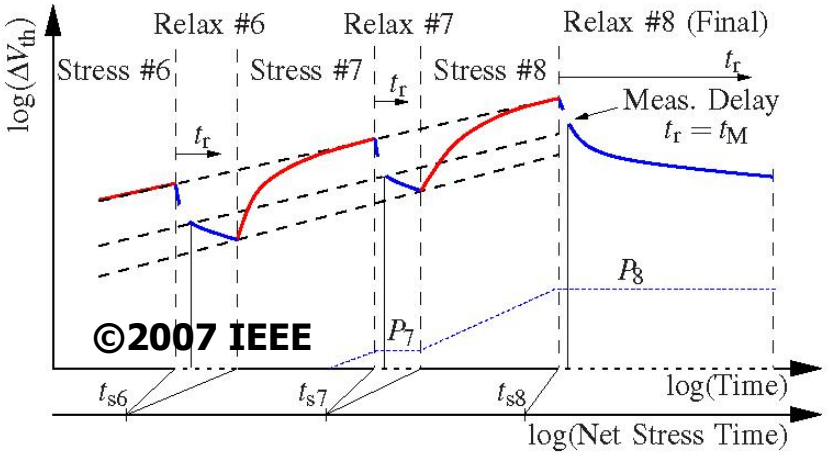


Figure 2.2: ΔV_{TH} decays once V_{GS} stress is removed. Accumulated V_{TH} changes have a quickly decaying, short-lived component, and a long-lived, or possibly permanent component. [28]

ΔV_{TH} is at least partially reversible. In the absence of stress, i.e. $V_{\mathrm{GS}} = 0$, ΔV_{TH} begins to quickly decay, as seen in figure 2.2. This recovery can be accelerated by applying an opposite voltage stress, for example, applying $V_{\mathrm{GS}} > 0$ to a pMOSFET. AC stress will thus result in a reduced ΔV_{TH} as compared with DC stress over the same usage time, though the degradation effect has a highly non-linear dependence on duty cycle. ΔV_{TH} is understood to comprise two components: a fast component which quickly begins to decay in the absence of stress (R), and a slower component which seems to be non-reversible and which plateaus

after a long time (P) [28]. The permanence of P is not clear, and it may actually decay at very long time scales [29].

2.1.2 Hot Carrier Injection

Semiconductor carriers which are significantly more energetic than the lattice at thermal equilibrium are called hot carriers. Short channel MOSFETs in saturation experience electric fields between the channel and drain which are large enough to produce hot carriers. These carriers are energetic enough to be injected from the channel into the gate oxide, leading to damage at the oxide interface and a gradual degradation of MOSFET parameters. This effect is known as hot carrier injection (HCI) [17]. The effect has a greater impact on nMOSFETs than pMOSFETs and is inversely related to channel length [30]. Modern MOSFETs operate with strong enough electric fields that HCI is an important concern in CMOS reliability [17], [18], [19], [21], [23], [31].

HCI effects can be regarded as a reduction in drain current, as expressed in (2.2) [18]. As in (2.1), there is an exponential dependence on temperature and a power law dependence on stress time with and eventual saturation in time. Unlike BTI, there is a strong power law dependence on V_{DS} rather than V_{GS}.

$$\Delta I_{\mathrm{D}} \propto I_{\mathrm{D}} \cdot V_{\mathrm{DS}}^{p} \cdot exp\left(\frac{-E_a}{kT}\right) \cdot t^{q} \tag{2.2}$$

Since I_d is a function of the overdrive voltage $(V_{\mathrm{G}} - V_{\mathrm{D}})$, HCI effects can alternatively be modelled as an increase in V_{TH} [17], [21], [23].

HCI in long channel devices has been described by the lucky electron model, where the driving force is the lateral electric field [30]. Under this model peak HCI degradation is correlated with peak substrate current, which is generally estimated to occur when $V_{\mathrm{GS}} = \frac{1}{2}V_{\mathrm{DS}}$. For modern technologies with channel lengths shorter than 250 nm, hot carriers are

better described by an energy driven model where the effect depends on the total available energy. Peak HCI degradation occurs when $V_{GS} = V_{DS}$ under this regime [30].

An attempt to separate the contributions of HCI and BTI to ΔV_{TH} found that HCI was responsible for almost half of the observed degradation [23]. BTI dominates performance losses in the short term. However, HCI has the larger time dependence and will come to be the dominant effect [19], [31].

HCI degradation persists in the absence of the responsible stress [32]. Thus an affected MOSFET will not begin to recover once it is powered off. However, partial recovery from HCI degradation is possible through elevated temperature annealing. Several studies have demonstrated this by heating MOSFETs either through special on-chip Joule heaters [32], or by forcing a large body-drain current [33]. At least once study suggests that partial recovery from HCI is possible by heating unpowered ICs in an oven for a short time [34].

2.1.3 Time Dependent Dielectric Breakdown

Time dependent dielectric breakdown (TDDB) is a sudden and irreversible localized change in a dielectric causing a discontinuous increase in conductivity [17], [19]. Stress conditions, primarily applied voltage and stress time, govern the generation rate of defects inside a dielectric. As the defects accumulate, the dielectric's properties gradually change until a breakdown occurs [17]. This dependence on defect generation means that TDDB is stochastic in nature, with the time to breakdown described by a Weibull distribution [17]. Aggressive CMOS scaling has led to gate dielectrics becoming thinner, reaching the 10 Å range. These dimensions ensure large electric fields across gate dielectrics, causing TDDB to be an important reliability concern [17], [19], [31], [35]. The introduction of high-k dielectrics has slowed the scaling of gate dielectrics, but it has not eliminated TDDB concerns [31].

In very thin dielectrics the time to breakdown is more difficult to identify than in thicker dielectrics. Breakdown is progressive and the current due to it increases incrementally rather

than sharply. Additionally, there is a larger current background noise due to quantum-mechanical tunnelling. These issues are highlighted in figure 2.3.

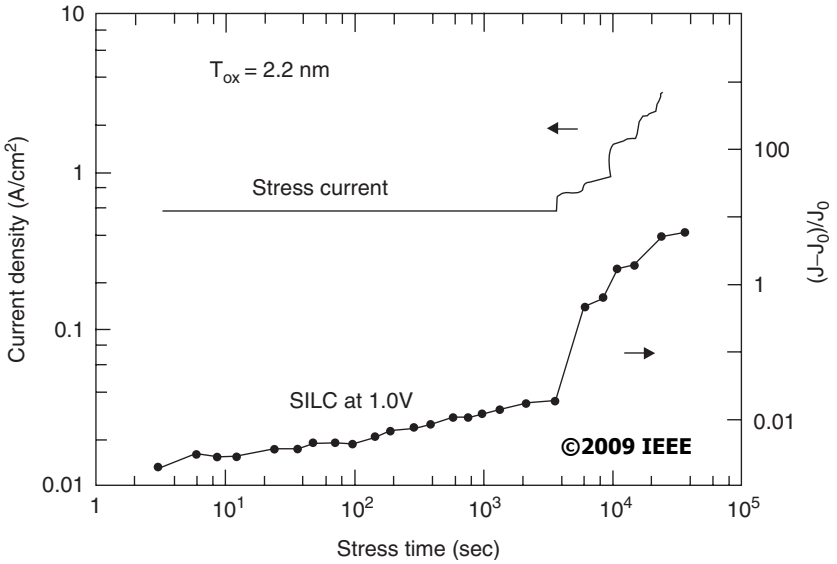


Figure 2.3: Pinpointing the time of breakdown in a thin dielectric is challenging. Breakdown current increases gradually and is partially masked by background quantum-mechanical tunnelling current. Stress-induced leakage current(SILC) measurements, where stress is temporarily removed and current is measured at a low voltage, can reveal breakdown effects [17].

Another issue with very thin dielectrics is that breakdown can occur in MOSFETs when the channel is not inverted (the device is off). This is a problem when a high bias is applied to the drain such as in burn-in screening, and certain I/O circuits [31].

2.1.4 Electromigration

Electromigration is the migration of atoms in a metal in response to an applied current. This effect does not manifest itself when the current density is low, as it is for example in household wiring. In contrast, the thin metal films present in ICs carry large enough densities for the effect to occur. This atomic migration is usually in the direction of electron flow, or the "electron wind". When enough atoms are displaced they can leave behind voids, leading to open-circuit failures. They can also pile up to form extrusions into the surrounding insulator, eventually reaching other lines thus causing short-circuit failures [17]. Electromigration

has long been considered a reliability issue for power nets, where unidirectional currents accentuated the effect. Aggressive CMOS scaling has caused current densities in signal nets to increase to the point that electromigration has become an issue here as well [19], [36]. Electromigration failures are statistical in nature. The median time to failure, the time for 50% of a population to have failed, is given by the empirical relationship known as Black's law:

$$t_{50} = A \cdot J^{-n} \cdot exp\left(\frac{E_a}{kT}\right) \tag{2.3}$$

where A is an empirical constant, J is the current density, n ranges from 1 to 2, E_a is an activation energy, and k and T are Boltzmann's constant and the temperature [17], [36].

A line's resistance will increase as electromigration progresses. The change will be hardly noticeable until the growing void reaches well over half of the line's width [17].

Aluminum (Al) was used exclusively for metallization for more than three decades. Today copper (Cu) has supplanted it thanks to its lower resistivity. The two metals' electromigration behaviours differ. Cu has much longer lifetime, about 100 times that of Al. Resistance in Cu increases in jumps rather than smoothly as in Al. Partly due to differing fabrication techniques, failure statistics for Cu have broader distributions than for Al [17].

2.2 Recycling Countermeasure Prior Art

Finding counterfeit ICs comes down to identifying those which have previously been in operation for any length of time beyond burn-in testing. In other words, the goal is to identify moderately aged ICs. There are two broad approaches to this problem, dubbed here as preemptive (section 2.2.1) and passive (section 2.2.2). Preemptive methods aim to ease counterfeit detection by modifying an IC design before fabrication, typically by including a specialized sensor. Such a strategy is not backwards compatible with already existing counterfeits, and may not be universally applicable. Passive methods seek to identify

counterfeits purely through non-destructive measurements performed on ICs. This approach is backwards compatible and potentially universal, but the tests may be onerous for an end-user to perform.

2.2.1 Preemptive Methods

The existing literature on preemptive methods can be broken down into three categories:

- Sensors that track the effects of physical degradation mechanisms such as BTI and EM (sections 2.2.1.1 and 2.2.1.2)
- Counters to track IC running time (section 2.2.1.3)
- Sensors that detect physical IC tampering (section 2.2.1.4)

2.2.1.1 BTI Sensors

Ring Oscillators for reliability As discussed in section 2.1.1, BTI is perhaps currently the largest concern in CMOS reliability. In response, a number of sensors have been proposed in order to track its effect on threshold voltage in order to increase IC reliability through improved design or adaptive performance [22], [23], [31], [35], [37], [38].

Ring oscillators (RO) are probably the most popular tool for tracking BTI-induced V_{TH} shifts [23], [31], [35], [38]. Unsurprisingly, this approach has been suggested for counterfeit detection [39], [40], [41]. The basic concept depends on the linear relationship between a RO's frequency and its transistors' threshold voltage [23]. Figure 2.4 shows the basic setup.

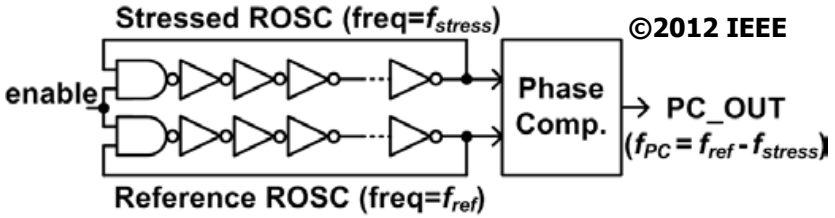


Figure 2.4: V_{TH} shifts over time can be tracked by comparing the frequency of a stressed ring oscillator to that of a reference ring oscillator [31].

Two ROs are built close to each other so that they will be matched. One RO, the reference, is power gated so that it is only powered when a reading from the sensor is needed. The reference transistors will therefore age very little and maintain a constant V_{TH} over the IC's life. The other RO, the stressed RO, functions continuously and its transistors age in a similar fashion to the IC's average transistor. By measuring the difference between the two RO frequencies, changes in V_{TH} can be tracked independently of local process and temperature variations [31].

Ring Oscillators for counterfeit detection While BTI reliability studies are concerned with devices near wearout, counterfeit detection is concerned with devices still firmly in their operating life. Sensors for counterfeit detection need to be more accurate, generating measurements with small variances. One group has addressed this with a sequence of increasingly capable sensors based on the architecture shown in figure 2.4 [39], [40], [41].

The first iteration, dubbed CDIR, is no different than a set-up used in reliability studies except for the addition of an integrated controller and frequency counter [39]. The controller allows each RO to be individually power-gated so that they can both be turned off during burn-in, and the stressed RO be turned on afterwards.

Due to their switching activity, transistors in a RO undergo BTI degradation only during half of the duty cycle and experience partial recovery during the other half of the duty cycle. The next iteration of the sensor, N-CDIR, aimed to subject the RO transistors to BTI stress during the full duty cycle [40]. As shown in Figure 2.5, transistors experience a constant voltage stress. A pass gate is placed at each inverter's output and a pull-down transistor at each inverter's input. This is done so that each pMOSFET can be put in a state of constant DC stress. When a measurement is needed, the pull-down transistors are shut off and the pass gates are opened to close the RO loop. The same architecture is used in both ROs in order to preserve matching.

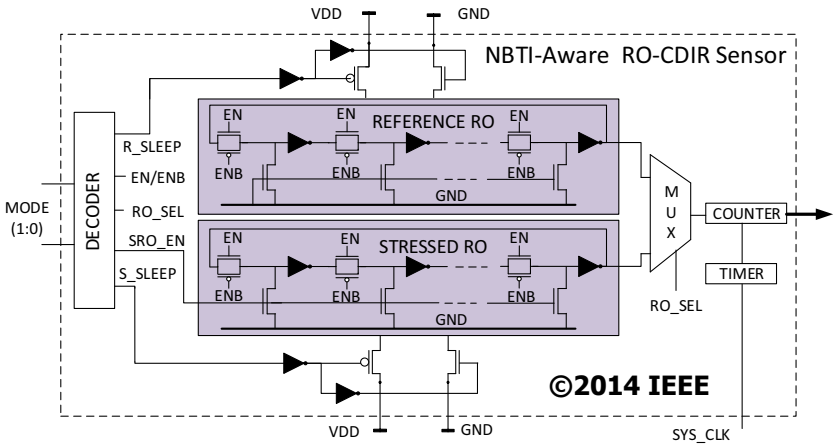


Figure 2.5: The N-CDIR sensor is designed to counteract BTI recovery [40].

Two schemes to improve upon N-CDIR have been proposed. The first, AN-CDIR, deploys an array of N-CDIR sensors in a single IC [41]. The average response is measured, leading to a reduction in the variance. The second, SN-CDIR, also deploys an array on N-CDIR [41]. In this case though each RO is individually addressable and only a subset of the "best" ROs will actually be used. The authors report a correlation between a RO's frequency when new and driven by a slightly higher voltage, and its frequency later in life when operating at a standard voltage. This correlation is used to select the stressed ROs predicted to experience the most dramatic aging. The reference ROs whose frequencies are closest to those of the stressed ROs at time 0 are then selected. The selection is programmed by burning fuses.

The authors tested their designs with Monte Carlo simulations carried out in Synopsys, Inc.'s HSPICE. Aging effects were simulated using HSPICE's aging simulator, MOSRA. 90 nm devices operating at 1.2 V were used with the three process variation profiles shown in table 2.1. The values in PV0 are claimed to be realistic without further justification. Simulation conditions and results for the various CDIR variants are summarized in table 2.2.

Though the reported results look impressive, they are likely optimistic. It is not clear why AN-CDIR and SN-CDIR used much larger ROs than CDIR. It is also unclear what process

Process Variations	Inter-die			Intra-die		
	Vth (%)	L (%)	Tox (%)	Vth (%)	L (%)	Tox (%)
PV0	5	5	2	5	5	1
PV1	8	8	3	7	7	2
PV2	20	20	6	10	10	4

Table 2.1: Process variations used in simulations of CDIR sensors [41].

Sensor	Age	Detection Error	Size	Process Variation	Monte Carlo instances
CDIR	1 month	0%	21 stages	PV0	10^3
CDIR	1 month	4.8%	21 stages	PV0	10^3
N-CDIR	3 days	Type I: 0.6% Type II: 0.45%	21 stages	PV0	10^3
N-CDIR	3 days	Type I: 10.19% Type II: 10.54%	21 stages	PV2	10^3
AN-CDIR	2 hours	Type I: 0.62% Type II: 0.81%	10 pairs 51 stages		10^3
SN-CDIR	2 hours	Type I: 0 Type II: 0.1%	6pairs 51 stages		10^3

Table 2.2: Performance simulations of CDIR sensors [39], [40], [41]

variances were assumed in these sensors. Nor is it clear how the error rate was determined to a precision better than 10^{-3} when all simulations used 10^3 Monte Carlo instances.

The differential nature of these sensors means that good matching between the reference and stressed devices is important. Properly matched transistors use a special layout to overcome local process and temperature variations The ROs will also need to contend with parasitic resistances and capacitances if oscillation frequencies are to be well matched. The required area overhead is certainly larger than what would be estimated from a gate count. All of this will make layout and placement a challenge. None of this was addressed.

Though BTI recovery was considered in the operation of N-CDIR, recovery during periods when an IC is powered down was not addressed. Some time, possibly years, elapses between recycled ICs' "deaths" and "resurrections". The ensuing BTI recovery would certainly have a significantly negative impact on an RO sensor's accuracy.

Interfacing with the sensor is also not addressed. If special-purpose test pins are to be avoided, the sensor's controller will have to be integrated into the rest of the IC's design. Special instructions will have to defined for controlling the sensor. This presents an added overhead for the IC designers both in terms of design and verification. Small machines as would be common in IoT may not have the ability to accommodate the inclusion of a new externally-accessible sub-system. The controller state will also need to be stored to ensure that it will remain in its normal mode except when a test is ordered. This means that a non-volatile memory will have to be included.

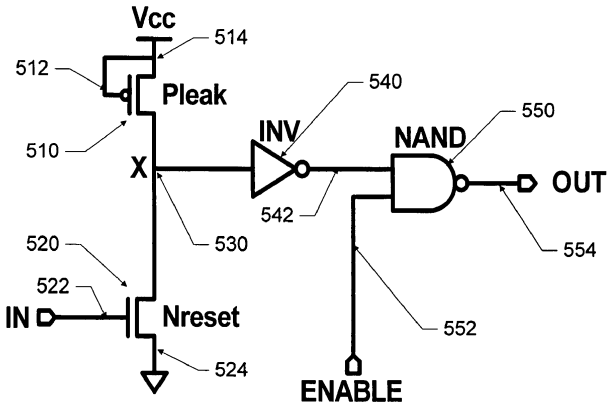


Figure 2.6: RO stage with high sensitivity to aging. Node X must be pulled up through P_{leak} which is permanently shut off [42].

Ring Oscillators in subthreshold By operating an RO in the subthreshold regime, its sensitivity to BTI can be boosted. This would allow the use of smaller structures, thus permitting easier matching and a smaller area overhead. The increased sensitivity is due to a MOSFET's drain current becoming proportional to $\exp V_{\mathrm{TH}}$ in the subthreshold, rather than $V_{\mathrm{TH}}^{\alpha}, \alpha \in [1, 2]$ as in triode and saturation regimes. Since switching response depends on I_{D} to charge a gate capacitance, a RO operating in the subthreshold should be very sensitive to changes in V_{TH}.

The RO stage shown in figure 2.6 uses sub-threshold behaviour to increase aging sensitivity [42]. The inverter's input X is pulled down through N_{reset} and pulled up through P_{leak}.

Since P_{leak} is configured to always be off, X will be pulled up by a leakage current. This sub-threshold behaviour is very sensitive to V_{TH} and hence to aging degradation effects on the RO will be magnified by 50 times as compared to a RO composed of simple inverters.

Another approach which exploits sub-threshold behaviour is presented in [35]. A regular RO is power-gated through a stressed pMOSFET. This transistor is biased so as to experience NBTI stress. When a measurement is needed it is biased in the sub-threshold regime, starving the RO of current. In this mode The RO's oscillation frequency is exponentially dependent on the power-gate MOSFET's V_{TH}. Experiments indicate a 53% shift in frequency for a 10% shift in V_{TH}.

Operating in the subthreshold regime leads to dramatically slowed circuit switching since I_D is several orders of magnitude below saturation levels. This would significantly lengthen measurement time and cause noise to become an issue. Device matching is also a challenge [27].

Wireless RO measurement Measuring a RO-based aging sensor's signal may be done wirelessly [43]. The focus of the cited work was to find a unique IC identifier to combat unauthorized production. The authors decided to use process variations as the basis of their work and to detect these variations with a ring oscillator. Their ring oscillator produced radio waves which were readily detected with a magnetic probe placed close to the IC's surface. This method would reduce the complexity of an aging sensor since no I/O pad would be needed to make measurements. One complication is that this measurement cannot be made with standard testing equipment.

2.2.1.2 Electromigration Sensors

Electromigration has been proposed as the basis for a counterfeit detection sensor [44], [45]. Rather than rely on a metal line's catastrophic failure, the idea is to track its change in resistance over time as it is continuously driven with a direct current. Such a structure is

much simpler and lighter weight than the RO based sensors described in section 2.2.1.1. The authors also claim an improved accuracy, though this is not convincingly demonstrated. The statistical nature of time to failure (TTF) of a line afflicted by electromigration presents a design challenge for the proposed sensor. Traditionally the effect was empirically modelled using Black's equation which lacks accuracy since it does not consider the effects of wire length and residual stresses from the manufacturing process. A more accurate physics-based model of electromigration is instead used in this work. Multiple parallel lines are used to increase the accuracy of the predicted TTF for the sensor. These lines are built close to each other in the same layer to minimize the influence of intra-die process variations. These parallel lines are isolated from each other and tied together through a different metal layer. The architecture of the sensor is shown in figure 2.7. The EM stressed wire set (EMS) is composed of multiple parallel lines and is powered by a DC current source so long as the IC is powered. The EM reference wire (EMR) is designed to have a predetermined voltage drop across it. The EMR's resistance remains constant since it is powered only sporadically when a sensor reading is needed. As the EMS ages, its resistance increases until the voltages across the two wire sets match and the output of the open-loop OPAMP (1-bit ADC) changes. The sensor is activated by the externally-applied signal `read_en` and the test result is stored in a 1-bit register. The authors suggest connecting the result register to an IC's JTAG circuits in order to make the sensor's output accessible.

Simulations were carried out by modelling the TTF with a lognormal distribution, which is commonly used to describe electromigration. The authors reported that an EMS composed of 6 lines is guaranteed to detect one year of aging to within 10 %, i.e. plus or minus about one month. A sensor of this size should be compact, presenting an area overhead of 0.02 % on a 25 mm^2 IC built in a 180 nm technology.

The reported accuracy performance of 10 % at one year appears to be an optimistic interpretation of the results. Figure 2.8, show a histogram of the age at which the sensor output changes state. The distribution is Gaussian-like with a mean and standard deviation

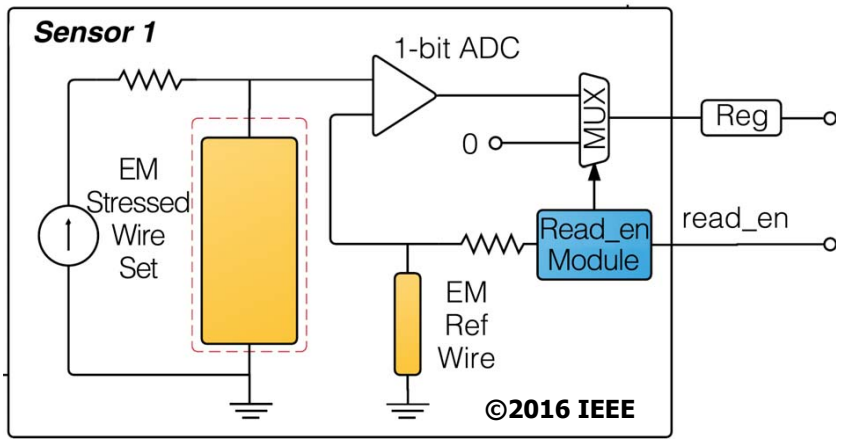


Figure 2.7: An aging sensor which exploits electromigration [45].

of approximated 1 year and 0.1 years respectively. Since only about 68 % of values fall within one standard deviation of the mean, the reported 10 % error cannot be regarded as correct. The results emerge from setting the electromigration induced lifetime of the wires to follow

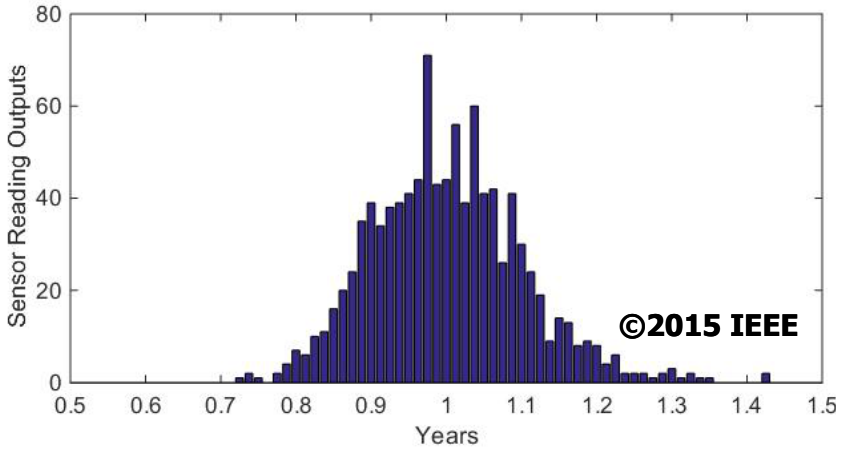


Figure 2.8: Histogram of ages when EM sensor output flips from 0 to 1 [44].

a lognormal distribution with a variance of 0.001. No justification is provided for this value. An added difficulty for this design may be the voltage and power requirements, neither of which are well addressed. It appears that a voltage in excess of V_{DD} may be required to induce accelerated electromigration. Power consumption may also be significant.

2.2.1.3 Life Counter

Recording IC activity with a programmable non-volatile memory has been proposed as a means of marking used ICs. The simplest idea is to count the system clock's pulses [46]. The proposed design uses two sequential counters to divide the clock frequency so that the time unit granularity and the maximum count can be adjusted for the available memory. Memory is written to sequentially so that the age of the system will be equal to the largest address which has been written to. The proposed resolution of one hour and maximum range of one year would require a 43 bit memory for a 2 GHz clock. The authors also suggest using the activity of rarely-switching nets instead of the system clock in order to reduce the memory requirements. In addition to the counters, the odometer design includes an adder to translate the pulse count to a memory address, a predefined non-volatile memory unit, and a controller for the odometer. On power-up, the controller searches the memory for the current IC age, outputs this information, then begins writing new data to the memory.

Well-known non-volatile memory technologies such as e-fuse and flash would not work well for such an application [47]. Though e-fuse is compatible with standard CMOS processes and widely used in analog-mixed signal applications, it must be programmed with pulses of several mA which cannot be done in the field. Embedded flash can be programmed in the field but is not compatible with standard CMOS. It requires up to 15 extra layers, making it uneconomical unless it comprises the majority of the IC's area. Instead the authors propose to use a newer technology, antifuse one-time programmable (OTP) memory, which is compatible with standard CMOS and is field-programmable. A downside of antifuse memory is that it is likely proprietary.

The advantage of this entirely digital approach to age measurement is a complete immunity against process variations. It also does not require any extra I/O pins as existing interfaces can be used to access read data. There are several drawbacks to this odometer idea. Using a net's switching activity as the "heartbeat" seriously complicates the design. Candidate nets must switch with the right frequency: not too frequently and not too infre-

quently. They must also switch consistently over the lifetime of the IC in all of its operating modes. Once selected, the signal must be routed to the odometer without impacting timing margins. This large and complex design presents challenges for real-world deployment in terms of increased cost, area constraints, and testing requirements. The most important concern though is the design's size and complexity.

A much simpler odometer is presented in [40]. Here a single bit of non-volatile memory in the form of a fuse records the state of the IC: new or used. The fuse controls the resistance in a test path, for example, between a special-purpose test pin and ground. Once the fuse is burned, the IC is permanently marked as used. A quick and simple continuity test will confirm the IC's status. The design is expensive to implement since it requires a special-purpose pin to access the fuse. The idea also suffers from having to rely on the system integrator to properly burn the fuse upon receipt of the IC. If the system integrator fails to do this for any reason, the security measure is made useless. Also, if there are several levels of system integration, everyone must implicitly trust the lowest-level integrator to not use counterfeit ICs.

2.2.1.4 Physical Tampering Sensors

An alternative strategy to tracking is to detect physical tampering of an IC. As discussed in Chapter 1, doing so through physical inspection is unsatisfactory. A more promising approach is presented in [48] where evidence of such tampering can be detected electrically. The sensor uses a pair of embedded flash cells for differential sensing. One of the cells is constructed normally, while the other has a floating gate which is not covered by the passivation layers. The unpassivated gate is exposed to the environment and changes in humidity, dust, static charge, and temperature will change the charge stored on it and thus be recorded. The sensor will function when the IC is powered down since flash is a non-volatile memory. Environmental anomalies recorded by the sensor are measured by comparing the read currents of the two cells. This would be especially useful for detecting if an IC has

previously been de-soldered and/or handled in a dirty environment. A demonstration chip was built in 350 nm process using a recently developed flash technology which is compatible with a standard CMOS process. To increase the sensor's sensitivity to temperature, the authors opened their chip's test enclosure. In a real situation, any IC would be packaged. It's unclear how sensitive this sensor would be to temperature spikes if it was encased in a plastic package. It's also unclear how a packaged sensor could be sensitive to changes in humidity or increased dust levels.

2.2.2 Passive Methods

Reported passive detection methods can be classified as:

- Remotely sensing emissions from an IC (section 2.2.2.1)
- Classifying path delays (2.2.2.2)
- Classifying parametric electrical measurements (2.2.2.3)

2.2.2.1 Remote Sensing

Recycled ICs may be detectable by light emissions during operation [49]. A small percentage of hot carriers will emit light in near infrared frequencies. These faint emissions are detectable during normal IC operation by single-photon detectors placed on the silicon backside. This is not a new idea. Measuring infrared emissions from switching events was developed by IBM in the early 1990s and is now widely adopted by test and failure analysis communities. Light emission due to off-state leakage current (LEOSLC) occurs when transistors are off, e.g. when a nMOSFET has its gate grounded and drain at V_{DD}. These emissions strongly depend on factors affecting the leakage current, including V_{TH}. Hence changes in V_{TH}, caused for example by aging, will manifest themselves as changes in LEOSLC intensity. LEOSLC can yield a great deal of information about an IC and has been used to extract functional blocks and perform reverse engineering. The proposed counterfeit detection scheme involves

measuring LEOSLC from several carefully chosen locations on an IC's backside. The measurement locations are selected so as to capture emissions from transistors with differing operating conditions such that some age quickly and others slowly. This differential measurement allows an IC to be its own reference, rendering inter-die variations inconsequential. The idea is interesting but it has two major drawbacks. First, measurements are made on the silicon backside, requiring the removal of the IC from its packaging, rendering the test *destructive*. Second, very detailed knowledge of the IC's layout and internal operation are needed for selecting the measurement sites, information that would have to be provided by the designer. This would make future proofing difficult in cases where documentation is lost or the designer goes out of business. It may also pose a security risk by aiding those seeking to reverse engineer the IC or carry out side-channel attacks against it.

2.2.2.2 Path Delay

IC path delays have been proposed as a basis for identifying recycled ASICs [50] and FPGAs [51]. MOSFET aging manifests itself most clearly as a reduction in I_{D}, which decreases switching speeds in a circuit since capacitances in the aged MOSFET's fanout now take longer to charge. The result is an increase in path delays, as is exploited by the RO-based sensors discussed in 2.2.1.1.

Path delay in ASICs A classifier trained on path delays was described in [50]. The delays in paths most susceptible to age-related slowdown were used as classification features. Such paths are associated with the following characteristics:

- Large numbers of XOR and NOR gates, which slow more rapidly than other gates.
- Experiencing many zeros, which cause pMOSFETs to experience a larger NBTI stress.
- Having a high switching probability, which causes a higher HCI stress on its transistors.

Classification was based on the delays of 100 paths: 50 with the most XOR and/or NOR gates, and 50 with either the most zeros or highest switching rates. Paths with many zeros or a high switching rate were determined through simulation with random input vectors. Path delays were measured with clock sweeping.

A simple 2-class classifier was trained on measurements of un-aged ICs. Two dimensional-reduction techniques were explored to reduce the 100-dimensional feature space. In the first, a single feature was randomly selected for retention. This was dubbed simple outlier analysis (SOA). In the second, principal component analysis (PCA) was used to reduce the space to 3 dimensions.

Data was generated by simulating three ISCAS'89 benchmark ICs using Synopsys' HSPICE. 300 Monte Carlo instances of each design were simulated in a 45 nm process with a 1.1 V supply voltage. Simulated aging was accomplished with HSPICE's MOSRA facility and set to 1 month, 3 months, 6 months, and 1 year. Three different process variation profiles resulting in V_{TH} inter-die variances of 3 %, 5 % and 8 % were used. Separate simulations were run for each V_{TH} variance.

Results for all 9 possible circuit/process variation combinations were not clearly reported. Table 2.3 summarizes the results for one particular benchmark circuit. These are representative of the rest of the work.

Method	1 month	1 year
SOA	50.7 % to 100 %	95.6 % to 100 %
PCA	72.7 % to 100 %	100 %

Table 2.3: Reported detection accuracy for the s38417 benchmark at two different ages and across all V_{TH} variances [50].

Though detection rates for one year old ICs seem promising, there are issues:

1. The training and testing data sets were identical. Excellent performance is the expected outcome of this situation but it provides no information on the generalizability of the classifier.

2. It is not clear if all of the selected paths were sensitized to a primary output. A path that can't control a primary output can't be measured.

3. Random workloads were used to age the test ICs. This may not be representative of a realistic workload.

4. Publishing the information end-users would need for testing may open an IC to side channel attacks or reverse engineering.

5. Required testing time was not addressed.

Path delay in FPGAs Path delay has also been proposed as a basis for detecting recycled field programmable gate arrays (FPGAs) [51]. These devices are composed of an array of cells which may be connected by programmable links. The cells are individually programmable and may implement almost any combinational function. A cell is configured by storing a truth table in its look up table (LUT), an SRAM array whose addressing is handled by a set of MUXs. A design is unlikely to use 100% of available cells, and the used cells are unlikely to have equal workloads. The authors therefore surmised that the distribution in LUT delays should change as an FPGA ages. They proposed to configure ring oscillators onto an FPGA in order to characterize the switching speeds of the device's LUTs. The ROs, shown in figure 2.9, can be set to use different paths through the LUTs.

The ROs consisted of 7 4-input LUTs chained together. One LUT input is reserved for connecting to the previous LUT's output, leaving 3 inputs available for 8 possible permutations. Oscillation frequencies for each permutation are measured and grouped into an 8-dimensional vector associated with the RO. Multiple ROs are deployed across an FPGA to classify it.

The RO data was then classified with two methods: a support vector machine (SVM), and k-means clustering. The SVM was used with two classes and was trained on measurements from un-aged FPGAs. Classification by k-means clustering was based on the supposition

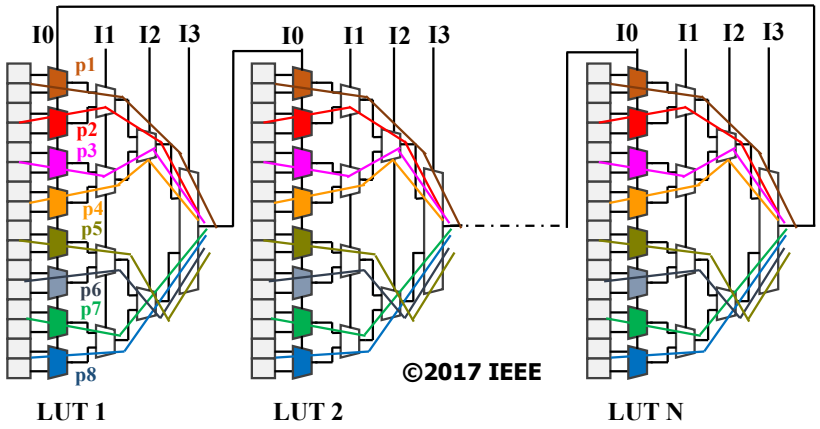


Figure 2.9: A Ring Oscillator composed of 4-input LUTs can be configured to use any one of 8 possible paths (p1-p8) [51].

that RO measurements should become increasingly multi-modal as an FPGA ages. A high number of clusters is therefore indicative of a used FPGA. The cluster count was selected to maximize the compactness of the clusters. A predetermined cluster number is used a threshold between new and recycled status.

Data was generated by simulating 4-input LUTs in a 90 nm process with 5% intra-die and 10% inter-die process variations. 30 "FPGAs", each comprised of 50 7-stage ROs were simulated. NBTI and HCI effects were simulated for ages of 2 hours, 1 day, 1 week, 1 month, and 6 months using HSPICE from Synopsys Inc. Aging was not uniformly applied to all paths in an RO in order to mimic a more realistic use case. Details of how this was done were not provided.

The training set for the SVM was composed of half (15) of the FPGAs before aging. The test sets were composed of the other 15 FPGAs aged for the above listed periods. It is implied that test sets consisted of equally aged devices. The authors claim a detection rate of 86% for one-day old devices, rising to 100% for 6-month old devices.

The k-means classifier was tested with data from all 30 FPGAs. The cluster number threshold was set to 3, no justification is provided. Detection accuracy is claimed to be about 60% for 1-day old devices and 100% for 6-month old devices.

The testing methodology is insufficiently described to ascribe meaning to the claimed detection accuracies. Specifically, the size, composition, and number of test data sets are not given.

This detection method may be quite slow in practice. Each suspect FPGA must be programmed and a number of ROs must be measured, with each one being measured several times.

2.2.2.3 Parametric Measurements

Ideally, the detection of recycled ICs should only depend upon measurements from standard parametric tests, which involve voltage and current sweeps, so as to avoid non-standard testing equipment.

Early failure rate analysis One group has proposed classification based on parametric measurements collected for early failure rate analysis [52]. These standard measurements are required before ICs can be released to market. The proposed classifiers are a SVM for individual ICs, a hypothesis test for batches of ICs, and a time-based parameter degradation analysis for individual ICs. Data used in this work came from real devices and was collected by Texas Instruments, Inc.

The SVM was implemented using the `LIBSVM` library and its decision boundary was set to the smallest hypersphere which could entirely enclose a training set consisting of ICs known to be new. The hypothesis test used Hotelling's two-sample t^2 statistic, which presupposes a Gaussian distribution [53], to compare a suspect batch to a set of known new ICs. The two-sided p-value was used with a significance level of 10 %.

The authors recognized that the accuracy of these two classification methods is greatly affected by inter-die process variations and sought a work-around. They did so by drawing on the work of others that showed that though aging degradation depends on both process variation and operating conditions, the degradation slope and its standard deviation are only

dependent on the operating conditions. Degradation curve sensitivity analysis (DCSA) seeks to identify counterfeit transistors from the slope of their degrading performance. Parameter degradation is modelled as a power law in time, resulting in a monotonic slope. Classifying ICs with DCSA involves measuring the degradation slope and determining if it exceeds a predetermined threshold taken from a reference curve measured during early failure rate analysis.

The concept of equivalent aging is central to both DCSA and early failure rate analysis. Since an IC's degradation, Δm, is a function of operating conditions, $\overrightarrow{s}$, and time, t, it is possible to map the degradation to different operating conditions:

$$\Delta m\left(\overrightarrow{s_1}, t_1\right) = \Delta m\left(\overrightarrow{s_{ref}}, t_{eq}\right) \tag{2.4}$$

An IC operating under $\overrightarrow{s_1}$ for time t_1 experiences the same degradation as if it had been operating under reference conditions $\overrightarrow{s_{ref}}$ for a time t_{eq}. This allows early failure rate analysis to study an ICs lifetime in a compressed time span. It is also used for slope measurements in DCSA. By applying the same stress conditions as during early failure rate analysis, slope measurements can be done quickly, and be mapped to the reference curve.

The authors tested their detection methods with three case studies: a DSP, a microprocessor, and a fully differential folded cascode amplifier. The DSP and microprocessor studies used measurements from real devices, whereas the amplifier data was simulated.

The DSP study used 35 devices randomly drawn from four lots. Two parametric measurements, F_{max} and V_{min}, were made at 7 logarithmically spaced point in time: $t_0 \ldots t_6$. The actual times were not given as this is industrially confidential information. The devices were considered new at t_0 and recycled at $t_1 \ldots t_6$. The SVM classifier's training set was formed from 18 randomly chosen devices at $t = t_0$. Classifier performance was measured with a cross validation protocol using the remaining 17 devices. Important points about the validation are:

- 7 validation sets, one per measurement time: $\{S_{V0}, S_{V1}, \ldots, S_{V6}\}$
- Classify 10 randomly chosen samples from each validation set
- Each validation set's classification rate came from averaging the classification results

The classification rate for the SVM was 30% at t_1, 80% at t_4 and 100% at t_6. The hypothesis testing method was also used with the same validation sets. Testing was done with randomly chosen groupings of 8 and 17 samples from each validation set, repeated 10 times. The classification rate was 100% at all measurement times. The hypothesis test was also performed for sample populations of randomly selected mixed-age devices. Group sizes ranged from 10 to 100 and the reported classification rate was a uniform 100%.

DCSA classification was based on F_{max}. One device was randomly chosen out of the 35 to build the reference curve. Since there were only 7 data points, the derivative curve was computed from an exponential fit to the data. The authors did not explain why they used an exponential function instead of a power law as they used when justifying DCSA. Validation used all measurements from the 34 remaining devices. For a threshold time of $t_{th} = t_1/2$, the reported classification rate was 100%. It is not clear how the slope was measured. According to the authors, DCSA needs two measurements closely spaced in time in order to estimate the aging slope. However, only 7 data points were taken for the DSP test devices and the spacing was logarithmic in time.

The microprocessor study used 313 devices randomly pulled from four lots. 49 parametric measurements were collected from various modules of each device. Measurements were made at 5 points in time. As with the DSPs, exact test times were not given. The authors previously presented these same results in an earlier publication where they used PCA to reduce the data's dimensionality to 9 from 49 [54]. Though it is not stated, I suspect that they have also done this in this paper. The authors used the same training and validation protocol as for the DSP study and achieved similar results. DCSA was based again on F_{max}. It was also performed as in the DSP study and achieved comparable results.

The final cast study was a fully differentiated folded cascode design from Texas Instruments. No physical devices were used, instead all data was from simulations. Four parameters were measured: gain, phase margin, bandwidth, and I_{ddq}. 100 Monte Carlo instances were seeded according to an unspecified process variation and aged with the RelXpert simulator from Cadence Design Systems, Inc. Data was collected at ages 0, 1,6, 12, 60, and 120 months. The SVD and hypothesis testing classifiers were trained and validated as with the other test cases, yielding similarly excellent results. The DCSA classifier was based on I_{ddq} data and used a threshold of 1 month. As with the other test cases, it performed without error.

Overall, this work's results are quite positive. The authors acknowledge a couple of limitations:

- The SVM and hypothesis test are susceptible to process variations.
- The training set corresponds to a specific set of process parameters. Devices produced under different conditions are likely to be classified with reduced accuracy.

The authors did not address the sensitivity of their classifier to process variations. The DCSA method depends on an accurate reference curve which may be expensive to construct due to the accelerated aging involved. There are several other issues in addition to what the authors have addressed. Results from the SVM classifier and the hypothesis test were presented together as results from a single once-class classifier. It is evident that the SVM classifier is less accurate. The hypothesis test may not perform as well as presented. The significance level was set to 10%, which is generally considered weak evidence at best[53]. More importantly, the validation sets used for the hypothesis test were composed *entirely* of aged devices. Doing so completely ignores the possibility that batches of suspect devices may contain some genuine components. The DCSA method requires end-users to perform parametric testing under specific conditions. This represents an added cost both in equipment and in time.

Self similarity Modern designs contain many self-similarities such as multi-core processors, systolic structures used in DSPs, and datapath elements. One group has proposed exploiting this in order to find both recycled components and hardware trojans (HTs) [55]. The basic idea is to compare the average switching current (I_{DDT}) of two identical modules while stimulated by the same test vectors. This allows detection without specialized on-chip sensors. The authors chose to base their work on I_{DDT} since it allows selective activation of smaller regions, and it is easier to generate test patterns than for path delays. The proposed method, dubbed self-similarity-based IC integrity analysis (SeMIA), is claimed to be robust against process variations and independent of a golden reference IC.

The switching power of two identical modules on the same chip should be correlated. The argument is: Test vector j is applied to module i on chip c to sensitize the gates in set $G_{i,j}$. The average I_{DDT} is then given by a power law:

$$I^{(c)}(G_{i,j}) = \sum_{g \in G_{i,j}} \beta_g \left(V_{DD} - \left(V_{\mathrm{TH}}^{(nom)} + \Delta V_{\mathrm{TH}}^{(c)} + \Delta V_{\mathrm{TH}}^{(c,g)} \right) \right)^{\alpha} \tag{2.5}$$

Where β_g is a gate-dependent constant, $V_{\mathrm{TH}}^{(nom)}$ is the nominal threshold voltage of the technology, $\Delta V_{\mathrm{TH}}^{(c)}$ represents the inter-die V_{TH} variation of chip c, and $\Delta V_{\mathrm{TH}}^{(c,g)}$ represents the intra-die V_{TH} variations of gate g on chip c. Intra-die variations are expected to be small and so can be linearized through a Taylor expansion:

$$I^{(c)}(G_{i,j}) = \sum_{g \in G_{i,j}} \beta_g \left(V_{DD} - V_{\mathrm{TH}}^{(nom)} - \Delta V_{\mathrm{TH}}^{(c)} \right)^{\alpha} - \sum_{g \in G_{i,j}} \beta_g \gamma_g \Delta V_{\mathrm{TH}}^{(c,g)} \tag{2.6}$$

Where $\gamma_g = \alpha \left(V_{DD} - V_{\mathrm{TH}}^{(nom)} - \Delta V_{\mathrm{TH}}^{(c,g)} \right)^{\alpha - 1}$. The average I_{DDT} of a separate, but identically designed, module i' is therefore related to that of module i:

$$I^{(c)}(G_{i',j}) = I^{(c)}(G_{i,j}) + S_{rand} \tag{2.7}$$

Where

$$S_{rand} = \sum_{g \in G_{i,j}} \beta_g \gamma_g \Delta V_{\mathrm{TH}}^{(c,g)} - \sum_{g \in G_{i',j}} \beta_g \gamma_g \Delta V_{\mathrm{TH}}^{(c,g)} \tag{2.8}$$

represents the effects of intra-die variations of V_{TH}. In a new and genuine IC, S_{rand} should be unbiased and in line with process variations. In a used IC, it is unlikely that two modules i' and i will have had the same usage history. If one module has been more heavily used, S_{rand} will become biased. Similarly, a HT would not be expected to be inserted into all modules on an IC. The added gates in the infected module will increase its switching current, effectively biasing S_{rand}. To characterize S_{rand}, a number of test vectors are applied and the resulting $\langle I^{(c)}(G_{i',j}), I^{(c)}(G_{i,j})\rangle$ pairs are fitted to a linear model. The residuals give estimates of S_{rand} for each test. The authors then compare the fit error to a predetermined threshold to decide if the suspect IC is in fact genuine or not.

The authors demonstrated SeMIA both in simulation and with measurements from an FPGA. Detection of both aged circuits and HT-infected circuits was attempted.

Aging effects were simulated with HSPICE MOSRA from Synopsys Inc. A 45 nm technology was used. V_{TH} was given a Gaussian distribution to model process variations. Inter-die variance was set to 15 % and intra-die variance was set to 10 %. Aging detection was tested in simulation using a 32-bit carry look ahead (CLA) adder and a 32-bit equality comparator. For both of these designs, it was assumed that the upper bits would be exercised less than the lower bits in real use cases. The upper halves of the circuits were therefore used as references. 200 Monte Carlo instances of each design were simulated, 200 instances underwent 1 year of simulated aging, and a further 200 instances underwent 5 years of simulated aging. Using SeMIA, the authors reported a detection error rate of 2.5% for the CLA adder and 2.25% for the equality comparator. The authors also demonstrated SeMIA on 5-stage processor, the DLX. They inserted the 32-bit CLA adder into the EX stage and were able to detect one year of aging with an error rate of 3.5%.

SeMIA was also demonstrated on physical FPGAs. 2 32-bit multipliers were deployed to different regions on an Altera DE0 board. The multipliers were exercised to different activity levels as they underwent accelerated aging by heating to 55 °C and boosting the supply voltage to 1.4 V. I_{DDT} was measured by placing a resistor across the board's power pin. Over 30 hours of aging, there was a clear difference in the two multipliers' currents, as seen in figure 2.10, providing good support for SeMIA. SeMIA's HT detection ability was

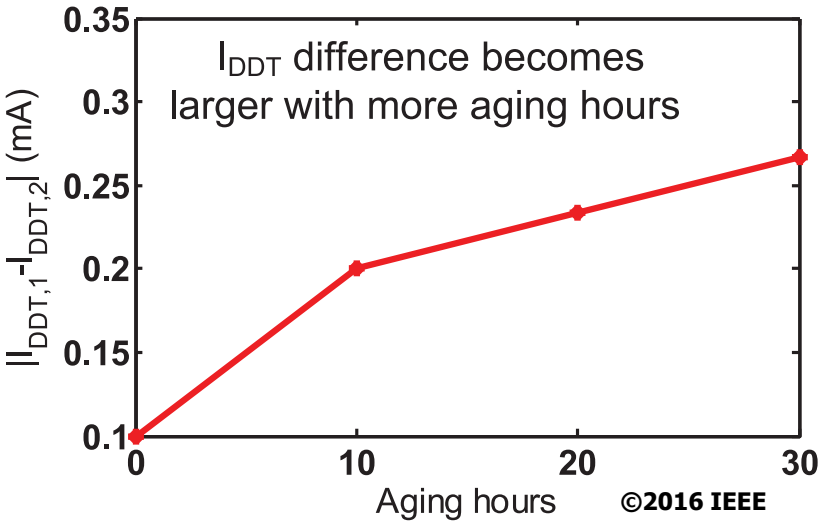


Figure 2.10: Difference in I_{DDT} for two 32-bit multipliers experiencing differential aging. Measurements taken from an Altera DE0. Accelerated aging conditions: T=55 °C, V_{DD}=1.4 V [55].

also explored. Small combinational HTs, including a 32-bit counter, were inserted into the test circuits. These HTs had an area overhead of about 2%. Detection rates were comparable to those for aged circuits.

SeMIA appears promising for detection of recycled ICs. However, the accuracy of the method depends upon the chosen fit error threshold level. It appears that the thresholds used in this work were learned from the entirety of the available data. Without proper validation, there is no way to assess the actual accuracy of the method. SeMIA seems to require many test points, this may take a while to execute.

As discussed in section 2.1, MOSFETs are subject to four major degradation mechanisms: bias temperature instability (BTI), hot carrier injection (HCI), time dependent dielectric breakdown (TDDB), and electromigration (EM). Detecting recycled ICs requires estimating IC age, which can be done by measuring MOSFET performance degradation. BTI and HCI both lead to a decrease in drain current over time; however, changes induced by BTI largely reverse themselves in the absence of stressing voltages. TDDB and EM permanently change gate impedance and metal trace resistance respectively. The changes are stochastic in nature and may be difficult to exploit for proper age estimation.

Proposed IC recycling countermeasures reviewed in section 2.2 fall into two categories: preemptive and passive. Preemptive methods involve modifications to an IC at design time in order to facilitate the identification of recycled ICs. These encompass on-chip sensors designed to exploit one of the above-mentioned degradation mechanisms, chronometers, and packaging anti-tampering mechanisms. Passive methods seek to identify recycled ICs purely through passive measurements without the need for design modifications. Proposed countermeasures suffer from a number of limitations. Proposed sensors all require extra communication interfaces, thus increasing IC design costs. These sensors target degradation mechanisms which are difficult to harness for proper IC age assessment. Proposed passive methods generally involve lengthy and complicated measurement regimes. Many also require that complex information be communicated to an end user in the form of a golden reference, or a trained pattern classifier. How this information is to be managed and communicated in trusted manner is not addressed. Some methods may also threaten the security of an IC designer's intellectual property by facilitating reverse engineering.

The next chapter lays out the arguments for a countermeasure in the form of an HCI-based sensor. The chapter includes an initial design study and promising preliminary performance estimates of such a sensor.

Chapter 3

Initial Design Study of a HCI-based IC Anti-Recycling Sensor

MANY countermeasures have been proposed for the problem of recycled ICs. The problem remains open as proposed solutions suffer from a number of issues, as discussed in chapter 2 and summarized here:

- Embedded sensors must communicate with the outside world and proposed designs do not have an elegant solution. Some require a dedicated pin, others propose to use a JTAG interface, others yet do not fully address the issue. If the JTAG interface is to be used, a controller for sensor is required. Some designs include a nonvolatile memory to keep track of an age counter. These features add complexity to the sensor and increase the cost and difficulty of integration into an IC design.

- The targeted degradation mechanism is difficult to harness for properly assessing the age of a sensor. For example, bias temperature instability begins to reverse once an IC is powered down, and electromigration is stochastic.

- Measurements may be complicated and lengthy. This is especially true for proposed passive methods such as measuring specific path delays.

- A golden reference device is required for a number of passive methods. Alternatively, complex information, such as path delays and corresponding test patterns or degradation curves, drawn from a golden reference are required. How this information is to be provided to an end-user is not addressed.

- Many passive methods relying on parametric measurements use a classifier to identify recycled ICs. However, the methods are sensitive to process variations and the classifier would need to be retrained any time there was a change in the production line. Who would retrain the classifier, and on which data is not addressed.

- Some methods, such as light emission due to off-state leakage current, can facilitate reverse engineering and pose a threat to the IC designer's intellectual property.

The above listed shortcomings all arise from the three main challenges to detecting a recycled IC:

1. a recycled IC is old but still operating within specs so signs of aging will be subtle;

2. a golden reference or some other reference is needed for comparison to identify an aged IC; and

3. measurement spread arising from fabrication process variations may mask signs of aging.

In trying to design an improved recycling countermeasure, a passive approach that depends on parametric measurements initially appears the most attractive. It has three attractive properties. First, no modifications to an IC design are required thus it presents no extra effort or cost to designers. Second, it is backwards compatible since it can be used with ICs which have already been produced. Third, measurements can be made using standard test equipment. However, there are challenging downsides. Identifying an IC-invariant feature, a measurement which can be used on any IC to judge its authenticity, is difficult and may

not be possible. This is not surprising since ICs span a wide range of designs, technologies, and usage patterns. Thus each IC requires its own measurement regime. Due to process variations, measurements will have a distribution and making a determination requires the use of statistical classification techniques. Thus each IC will require its own classifier which has been trained on a set of golden reference devices. Who is responsible for training these classifier is an open question, as how they should be distributed to end-users. Furthermore, due to sensitivities to process variations, the classifiers would likely need to be retrained whenever changes were made to production lines.

Despite the added overhead it presents for an IC designer, an embedded sensor is an overall better strategy. Because it is designed for the purpose of a countermeasure, an embedded sensor can generate a clear and easily measured aging signal. Since the same sensor can be deployed in different IC designs, its signal can be an IC-invariant feature. It can have its own internal reference, eliminating the need for a golden reference. By employing matching between the reference and aging devices, measurement spread due to process variations can be reduced. The drawback of an added overhead during the IC's design can be minimized with an improved sensor design.

Though multiple on-chip anti-recycling sensors have been proposed, there is room for significant improvement. In this chapter's first section, the features of an improved anti-counterfeiting sensor are considered. In the second, an initial feasibility study is presented.

3.1 Design Considerations

An anti-recycling sensor needs to track an IC's performance degradation over time and to do so consistently. The effect it tracks should be easy to measure, easy to control, and difficult or expensive to undo. To be realizable, the sensor will need to be compatible with a standard CMOS process without violating design rules. To be useful it will have to accurately

measure an age of 6 months or better. It should also have a minimal footprint so that it can be embedded into an IC design with minimal disruption.

3.1.1 Aging Mechanism

Of the four CMOS degradation mechanisms described in section 2.1, HCI best fits the requirements of an on-chip anti-recycling sensor.

Time dependent dielectric breakdown causes a permanent drop in a MOSFET's gate impedance. However, in thin oxides the change is gradual rather than sudden and so may be difficult to measure. Also, the stochastic nature of the effect would add a source of measurement uncertainty in addition to that due to technological process variations. The voltage required to induce an oxide breakdown in a reasonably short time may also be impractically large for an on-chip design.

Like time dependent dielectric breakdown, electromigration is a permanent change and would make sensor tampering difficult or impossible. As with time dependent dielectric breakdown, electromigration is also stochastic in nature. As covered in section 2.2, an electromigration-based sensor has been proposed. It is not clear how well the design overcomes the difficulties in estimating sensor age that are incurred by the mechanism's stochastic nature. The error in age estimation appears to be understated by a factor of three. Voltage and power requirements, which may be significant, are also not properly addressed.

The significant work done to characterize bias temperature instability effects for reliability studies provides an easy base to build upon, as seen in section 2.2. There is however a significant drawback to basing a recycling countermeasure on bias temperature instability: the resulting effect is largely reversible. So long as an IC is powered and functioning, bias temperature instability will degrade MOSFET performance. However, as discussed in 2.1, bias temperature instability effects begin to revert when voltage stress is removed, i.e. when the chip is powered down. A counterfeit IC can be expected to spend at least several months,

if not much longer, without power until it is resurrected and redistributed. This relaxation would certainly make proper age estimation very difficult.

HCI is the most promising degradation phenomenon for use in an anti-recycling sensor. Like bias temperature instability, it is modelled as a power law in time and thus easier to measure than a stochastic mechanism. The magnitude of the effect is a function of geometry and applied voltage; it is therefore easily controlled. Because it affects MOSFETs as opposed to metal traces, HCI can be tracked using a much smaller structure than time dependent dielectric breakdown. Unlike bias temperature instability, the effects of HCI do not relax in the absence of applied voltages and will not reverse once an IC is powered off. HCI-induced changes can be partially reversed through high temperature annealing. Despite this, exploiting HCI presents a fruitful strategy for combatting recycling. Circumvention of an HCI-based countermeasure entails additional costs in terms of equipment, process development, and crucially extra testing. Identifying scrap ICs suitable for recycling entails a certain amount of functional testing. The presence of a countermeasure threatens to significantly increase this testing effort as any annealed IC would need to be retested before being sold. Any notable increase in the cost of counterfeiting significantly lessens its appeal.

3.1.2 Sensor Signalling

A number of proposed sensors described in section 2.2 express aging as a frequency shift. The simplest such structure is a ring oscillator whose frequency is externally measured. The generated frequency tends with the inverse of the size of the oscillator so there is a cost trade-off between sensor area overhead and measurement equipment. The signal will either be cheap to generate and expensive to measure, or expensive to generate and cheap to measure. Ensuring an easier measurement by for example, including an on-board frequency counter, adds area overhead and complexity. Added overhead complicates integration, potentially significantly so.

A DC quantity can be easily and quickly measured with inexpensive equipment. Without the need for oscillators or frequency counters, the sensor can be kept quite small. Thus in the case of generating and measuring a DC quantity, there is no significant trade-off between the implementation and measurement costs.

3.1.3 Measurement Interface

The simplest way to access an embedded aging sensor is to provide it with a dedicated interface, including single-purpose pins. Doing so significantly adds to the development and production costs of an IC, creating a serious roadblock to adoption. A more cost-effective strategy is to re-purpose one of the IC's existing interfaces. This may involve integration into the IC's JTAG circuit as in [44], [45], or "borrowing" an output pin during a special test mode. Depending on the information to be read from the sensor as well as the details of the interface, this approach may still be too complex to be easily implemented. For example, even a simple state machine would present added area overhead which must be properly placed in an existing design as well as an added testing burden.

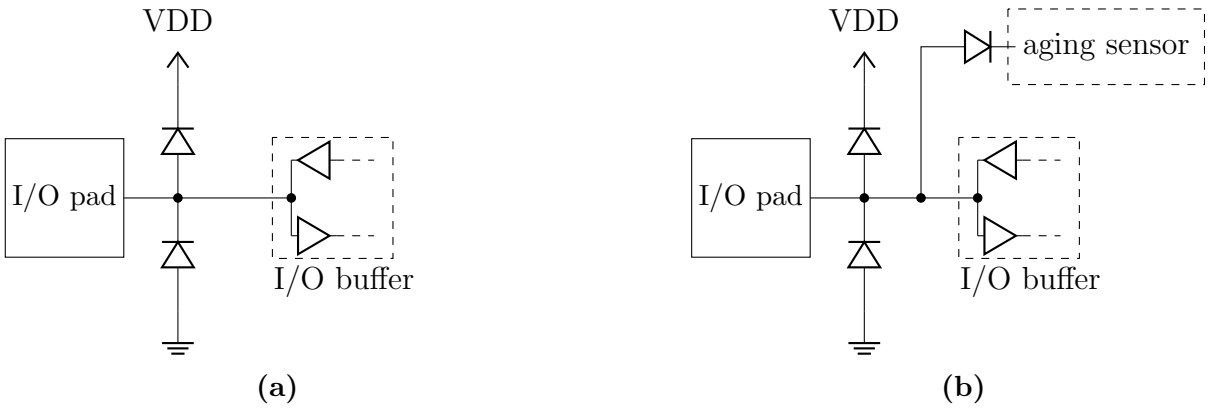


Figure 3.1: (a) Typical electrostatic discharge (ESD) protection scheme for an IC's I/O [27]. (b) Possible analog interface for a sensor which reuses the ESD protection without interfering with the I/O functionality.

A sensor which produces a DC signal could be accommodated with a simple analog interface. Figure 3.1 presents such a possible interface. Figure 3.1(a) shows a standard

electrostatic discharge (ESD) protection used to protect CMOS ICs from large input voltages which can damage MOSFET gates. In the proposed interface, seen in Figure 3.1(b), the sensor effectively controls the input impedance when a sufficiently large voltage is applied to a pin. Under normal operating conditions the sensor is isolated from the I/O interface and does not interfere with its operation. This very simple set-up would be best used with a parametric measurement and provides several benefits:

- **No interference with the rest of the IC:** The sensor and the measurement interface do not need to be functionally integrated into the IC design. When the IC is operating under normal conditions, the sensor becomes entirely invisible and inaccessible.
- **Easy to test:** There are no state machines to verify. A manufacturer can easily test the sensor by performing a parametric test.
- **Easy to use:** Applying a large enough voltage gives access to the sensor. Depending on the sensor design, a measurement may be as simple as a DC current measurement.

3.1.4 Measurement Accuracy

There are two measurement challenges when determining an IC's authenticity via an on-chip sensor: obtaining an accurate sensor measurement and comparing that measurement to an accurate reference.

Sensor measurement variance arises from manufacturing process variations. These are classified into two components: intra-die or local variations, and inter-die or global variations. From the law of large numbers, measurement variance can be lessened by averaging over multiple samples [56]. Device matching, a set of layout techniques intended to minimize differences between specific devices on the same die, employs spatial averaging to reduce local variation effects. Strategies such as placing devices symmetrically around a region centroid, using multiple transistors in parallel, or using larger transistors minimize the influence of process gradients over the die [27]. Averaging can also be performed over multiple sensors,

either on the same IC as in [41], or across multiple ICs. Deploying multiple sensors on the same IC would serve to reduce local variation effects, which can be useful for more accurately measuring the age of that single IC. An end-user is likely to want to authenticate a batch of ICs obtained from a vendor. In such a scenario, exhaustive testing of each IC is highly inefficient as compared to random sample testing. By averaging over a sufficiently large sample, global variation effects can be controlled.

The reference can be either sourced externally or generated internally. In the case of an external reference, it must be provided to an end-user by the IC manufacturer. If the reference information is simple and concise, it may be included into a data sheet easily enough. The advantage of an external reference is a minimized sensor since no reference value need be produced. The disadvantage, in addition to the need to communicate a reference value, is that measurements are subject to the full effects of process variations. However, this may not be serious impediment to accurate measurement if batch testing of ICs is employed. In the case of an internal reference, the sensor generates its own reference signal. Local variation effects can be minimized by employing matching techniques between the reference and aging parts of the sensor. There is also no need for the IC manufacturer to communicate reference values to end-users. These advantages come at the price of a larger overhead for the sensor.

3.2 Initial Performance Estimates

As discussed above in section 3.1, an embedded anti-recycling sensor which exploits HCI and outputs a DC signal could provide advantages over other proposed approaches. In this section a theoretical proof-of-concept sensor with these features is modelled to determine its ability to generate a detectable aging signal by a reasonable age. Globally, consumers keep their mobile phones on average for more than 20 months [57]. Using this as a proxy for the age of recycled components, such counterfeits can reasonably be expected to range in age

from months to years. Thus the age threshold by which a recycled IC should be detectable can be safely set at 6 months.

Detailed circuit simulations were not performed at this point since the goal was only an initial assessment of HCI degradation of the sensor. Deployment in a real 65 nm technology from TSMC [58] was considered and relevant parameters were used in the modelling.

This proof-concept sensor consists of a matched pair of nMOSFETs: M_{age}, which is exposed to elevated DC voltages such that it experiences accelerated HCI degradation, and M_{ref}, which acts as a reference. As the sensor, and the IC in which it is embedded, ages the threshold voltage of M_{age}, $V_{TH,age}$, increases. Thus the differential threshold voltage $\Delta V_{TH} = V_{TH,age} - V_{TH,ref}$ increases monotonically as time progresses. It consists of two components: the HCI contribution, dV_{TH}, and the intrinsic mismatch between the transistors, δV_{TH}. For the proof-of-concept to be considered effective, dV_{TH} must be distinguishable from δV_{TH} within 6 months.

Due to local process variations, even matched transistors have an intrinsic mismatch. δV_{TH} is modelled in the first order as following a Gaussian distribution with a mean of zero and a standard deviation of

$$\sigma_{\delta V_{TH}} \approx \frac{A_{VTH}}{\sqrt{W \cdot L}}, \tag{3.1}$$

where W and L are the nominal channel width and length, and A_{VTH} is a technology dependent parameter [59].

The HCI-induced V_{TH} shift in an nMOSFET can be described by the compact model [60]:

$$dV_{\mathrm{TH}}(t) = A_{\mathrm{HCI}} t^{\frac{1}{1+n_x}} \tag{3.2}$$

$$A_{\mathrm{HCI}} = C_{\mathrm{HCI}} \left[\left(V_{\mathrm{GS}} - V_{\mathrm{TH},0} \right) K_\nu \right]^{\frac{n_x}{1+n_x}} \left(\frac{n_x}{L} \right)^{\frac{1}{1+n_x}} \tag{3.2a}$$

$$K_\nu = \exp\left(\frac{E_{\mathrm{ox}}}{E_0} \right) \exp\left(\frac{-\phi_{\mathrm{IT}}}{q\lambda E_{\mathrm{lat}}} \right) \exp\left(\frac{E_{\mathrm{a}}}{kT} \right) \tag{3.2b}$$

$$E_{\mathrm{ox}} = \frac{V_{\mathrm{GS}} - V_{\mathrm{TH},0}}{t_{\mathrm{ox}}} \tag{3.2c}$$

$$E_{\mathrm{lat}} = \frac{V_{\mathrm{DS}} - V_{\mathrm{DSAT}}}{l} \tag{3.2d}$$

$$V_{\mathrm{DSAT}} = \frac{E_{\mathrm{sat}} L \left(V_{\mathrm{GS}} - V_{\mathrm{TH},0} \right)}{E_{\mathrm{sat}} L + \left(V_{\mathrm{GS}} - V_{\mathrm{TH},0} \right)} \tag{3.2e}$$

$$E_{\mathrm{sat}} = \frac{2 v_{\mathrm{sat}}}{\mu_{\mathrm{eff}}} \tag{3.2f}$$

$$\mu_{\mathrm{eff}} = \frac{\mu_0}{1 + \theta \left(V_{\mathrm{GS}} - V_{\mathrm{TH},0} \right)}. \tag{3.2g}$$

This model estimates dV_{TH} over time under constant voltage stress. It is a function of time, t; gate-source voltage, V_{GS}; drain-source voltage, V_{DS}; and channel length, L. It has been experimentally verified against an IMEC 65 nm process using the parameter values listed in Table 3.1 [60]. The published model was reused here with slight modifications. TSMC-specific values for the oxide thickness, t_{ox}, and the initial threshold voltage, $V_{\mathrm{TH},0}$ were used instead of those listed in Table 3.1. Table 3.2 lists the new values.

Table 3.1: (3.2) parameter values for an IMEC 65 nm process [60].

$n_x = 1.21$	$C_{\mathrm{HCI}} = 1.5 \times 10^{-5}$	$E_0 = 7.1 \times 10^9\,\mathrm{V\,m^{-1}}$
$\phi_{\mathrm{IT}} = 3.7\,\mathrm{eV}$	$\lambda = 7.8\,\mathrm{nm}$	$E_{\mathrm{a}} = -6 \times 10^{-2}\,\mathrm{eV}$
$V_{\mathrm{TH},0} = 200\,\mathrm{mV}$	$t_{\mathrm{ox}} = 2.2\,\mathrm{nm}$	$l = 45\,\mathrm{nm}$
$v_{\mathrm{sat}} = 1 \times 10^5\,\mathrm{m\,s^{-1}}$	$\mu_0 = 235\,\mathrm{cm^2\,V^{-1}\,s^{-1}}$	$\theta = 0.95\,\mathrm{V^{-1}}$

Two design cases of the proof-of-concept sensor were examined: using core nMOSFETs and using I/O nMOSFETs. For each, four biasing scenarios were explored:

- B0: $V_{\mathrm{GS}} = V_{\mathrm{DS}} = V_{\mathrm{DD}}$. Both gate and drain voltages are set to the nominal supply voltage. This is the base case and no significant HCI degradation is expected.

Table 3.2: Device parameters for a standard V_{TH} core and a I/O nMOSFET from a 65 nm TSMC technology [58].

nMOSFET	**min L (µm)**	**min W (µm)**	**t_{ox} (nm)**	**$V_{TH,0}$ (mV)**	**V_{DD} (V)**
core	0.065 ± 0.004	0.6 ± 0.008	2 ± 0.06	171	1.0
I/O	0.28 ± 0.008	1 ± 0.012	3.4 ± 0.13	370	1.8

- B1: $V_{GS} = V_{DS} = 1.1V_{DD}$. V_{GS} is set to the maximum recommended value [58], and V_{DS} is set to match. A core device is expected to experience a greater degradation compared to an I/O device since as discussed in section 2.1, equal gate and drain biasing causes peak HCI in channel lengths below 250 nm.

- B2: $V_{GS} = V_{DD}, V_{DS} \approx 2V_{DD}$. Voltages can be roughly doubled with the use of a charge pump. Here V_{DS} is set to $0.9 \cdot 2\,(V_{DD} - V_{TH})$, 90 % of the theoretical maximum output of a simple charge pump [27]. V_{GS} is kept at its nominal value to avoid risking damage to the gate oxide. Greater degradation is expected in the I/O device under this biasing condition since the channel length is above 250 nm.

- B3: $V_{GS} = \frac{1}{2}V_{DD}, V_{DS} = V_{DD}$. No significant HCI degradation is expected due to the lowered drain current relative to scenario B0. However, this biasing may cause a larger HCI degradation in the longer channel I/O device since $V_{GS} = \frac{1}{2}V_{DS}$.

Initially M_{age} has not yet experienced any HCI degradation and $dV_{TH} = 0$. As HCI begins to degrade M_{age}, dV_{TH} increases and will eventually become large enough to be distinguishable from δV_{TH}. Since $\mu_{\delta V_{TH}} = 0$, this might intuitively be expected to occur once $\mu_{dV_{TH}} \geq \sigma_{\delta V_{TH}}$. If $\mu_{dV_{Th}} = \sigma_{\delta V_{TH}}$, it would be larger than 84 % of the expected possible δV_{TH} values. At $3\sigma_{\delta V_{TH}}$ this would be 99 %.

As a first performance estimate, $\sigma_{\delta V_{TH}}$ was compared to $\mu_{dV_{TH}}$ after six months of aging. Nominal device parameter values (Table 3.2) were used to calculate both $\sigma_{\delta V_{TH}}$ and $\mu_{dV_{TH}}$. A value of 3 µm mV was used for A_{VTH} in (3.1) [60]. Results are summarized in Table 3.3. For easier interpretation of the results Table 3.4 normalizes $\mu_{dV_{TH}}$ in terms of $\sigma_{\delta V_{TH}}$.

Table 3.3: Comparison of mean HCI-induced V_{TH} shift to intrinsic V_{TH} mismatch due to local process variations. Predictions are for 6 months of aging under four different biasing conditions.

		$\boldsymbol{\mu_{dV_{\mathrm{TH}}}}$ **(mV) @ 6 months**			
nMOSFET	$\sigma_{\delta \mathbf{V_{TH}}}$ **(mV)**	**bias B0** $\mathbf{V_{GS} = V_{DD}}$ $\mathbf{V_{DS} = V_{DD}}$	**bias B1** $\mathbf{V_{GS} = 1.1V_{DD}}$ $\mathbf{V_{DS} = 1.1V_{DD}}$	**bias B2** $\mathbf{V_{GS} = V_{DD}}$ $\mathbf{V_{DS} \approx 2V_{DD}}$	**bias B3** $\mathbf{V_{GS} = \frac{1}{2}V_{DD}}$ $\mathbf{V_{DS} = V_{DD}}$
core	15.2	0.002	0.030	55.4	0.010
I/O	5.67	0.062	0.340	875	36.8

Table 3.4: Mean HCI-induced V_{TH} shift expressed in terms of intrinsic V_{TH} mismatch due to local process variations. Predictions are for 6 months of aging under four different biasing conditions.

	$\boldsymbol{\mu_{dV_{\mathrm{TH}}}/\sigma_{\delta V_{\mathrm{TH}}}}$ **@ 6 months**			
nMOSFET	**bias B0** $\mathbf{V_{GS} = V_{DD}}$ $\mathbf{V_{DS} = V_{DD}}$	**bias B1** $\mathbf{V_{GS} = 1.1V_{DD}}$ $\mathbf{V_{DS} = 1.1V_{DD}}$	**bias B2** $\mathbf{V_{GS} = V_{DD}}$ $\mathbf{V_{DS} \approx 2V_{DD}}$	**bias B3** $\mathbf{V_{GS} = \frac{1}{2}V_{DD}}$ $\mathbf{V_{DS} = V_{DD}}$
core	1.56×10^{-4}	1.98×10^{-3}	3.65	6.58×10^{-3}
I/O	1.09×10^{-2}	6.00×10^{-2}	154	6.49

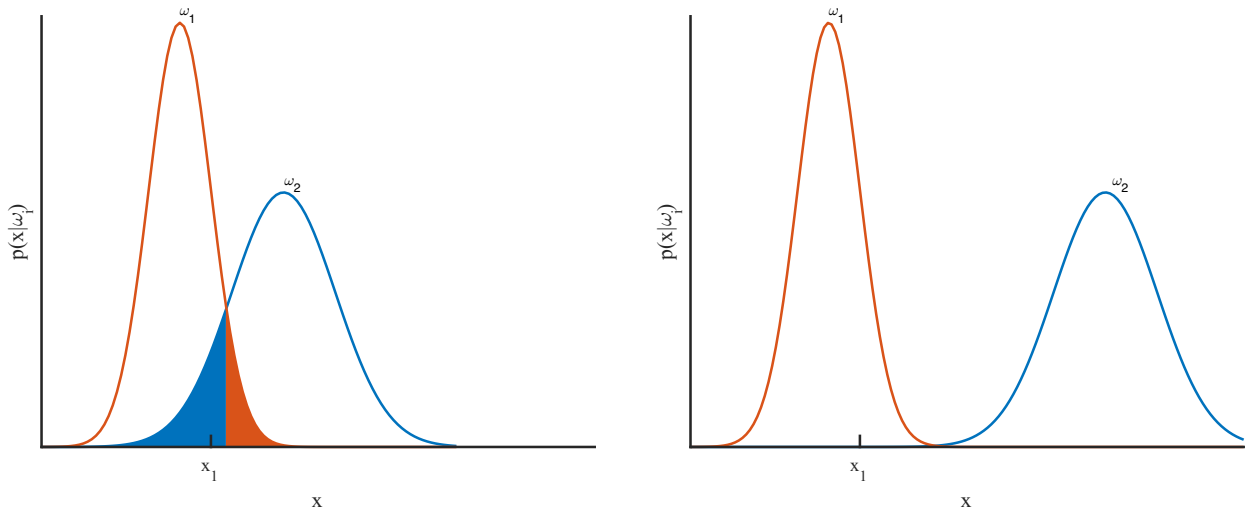
Bias B2 results in a significant HCI degradation in both devices. This is also the case for the I/O device with bias B3. These results appear very promising but must be treated with some caution. This analysis assumed that $dV_{\mathrm{TH}} = \mu_{dV_{\mathrm{TH}}}$, when in actuality dV_{TH} is a random variable since each nMOSFET is unique due to process variations.

A better estimate of the proof-of-concept sensor's performance can be made by taking this variability into account. The problem of distinguishing between dV_{TH} and δV_{TH} based on a measurement can be analyzed through the framework of Bayesian decision theory [61]. The ease of distinguishing between dV_{TH} and δV_{TH} is characterized by the Bayes error, the probability of confounding the two. The concept is illustrated in Figure 3.2. Consider a situation consisting of two possible states ω_1 and ω_2, each described by a probability distribution $p(x|\omega_i)$. Bayesian classification exploits Bayes' theorem to determine which state is most likely to have occurred based on a measurement x. Under the assumption of equal a priori probabilities, that the states are considered equally likely before knowing x, the Bayes' decision rule is simply $\arg\max_j p(x|\omega_j)$. When the two distributions are widely spaced, as in Figure 3.2(a), the probability of making an error is remote. Here the measurement x_1 is easily classified as being due to ω_1. In Figure 3.2(b) the distributions are partially overlapped and x_1 may be misclassified. The integral of the overlapping area, shown as the shaded region, is the probability of making an erroneous classification and is known as the Bayes error.

In the case at hand, the two states are

- ω_1 : There has been no HCI degradation and $\Delta V_{\mathrm{TH}} = \delta V_{\mathrm{TH}}$,
- ω_2 : HCI degradation has occurred and $\Delta V_{\mathrm{TH}} = \delta V_{\mathrm{TH}} + dV_{\mathrm{TH}}$.

The distribution $p\left(\Delta V_{\mathrm{TH}}|\omega_1\right)$ is Gaussian with a mean of zero and standard deviation $\sigma_{\delta V_{\mathrm{TH}}}$. The distribution $p\left(\Delta V_{\mathrm{TH}}|\omega_2\right)$ is also assumed to be Gaussian with a mean of $\mu_{dV_{\mathrm{TH}}}$ and standard deviation $\sqrt{\sigma^2_{\delta V_{\mathrm{TH}}} + \sigma^2_{dV_{\mathrm{TH}}}}$, which depends on the following assumptions:



(a) **(b)**

Figure 3.2: (a) Bayes error is small when distributions are widely spaced. (b) Bayes error grows as the overlap in distributions increases.

1. dV_{TH} has a Gaussian distribution. This is reasonable because nothing is known about the actual distribution and the Gaussian distribution is the maximum entropy continuous distribution [61].

2. δV_{TH} and dV_{TH} are independent. This is a gross simplifying assumption which may be made since the goal is not to derive a highly accurate performance model, but to assess in a first order whether an HCI effect may be detectable. In actuality there is no independence since both δV_{TH} and dV_{TH} depend on transistor geometry.

The standard deviation of dV_{TH} was approximated by including the tolerances on L and t_{ox} listed in Table 3.2 when calculating dV_{TH}:

$$\sigma_{dV_{\mathrm{TH}}} \approx dV_{\mathrm{TH}}\left(t, V_{\mathrm{GS}}, V_{\mathrm{DS}}, L - \Delta L, t_{\mathrm{ox}} - \Delta t_{\mathrm{ox}}\right) - dV_{\mathrm{TH}}\left(t, V_{\mathrm{GS}}, V_{\mathrm{DS}}, L, t_{\mathrm{ox}}\right). \tag{3.3}$$

Figure 3.3 shows the estimated Bayes error when distinguishing an HCI effect from a differential threshold voltage measurement of the proof-of-concept sensor. The results are very promising. By six months, the error for a sensor based on I/O nMOSFETs and operating under bias condition B2 or B3 is inconsequential. For a sensor based on core nMOSFETs

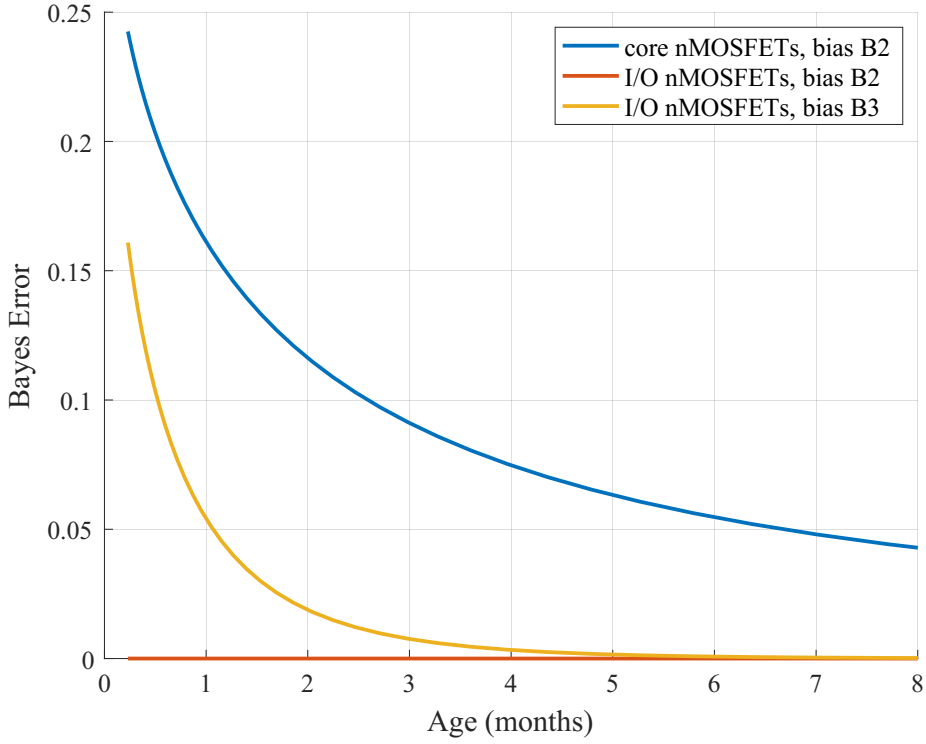


Figure 3.3: Bayes error when detecting an HCI effect from a differential threshold voltage measurement.

and operating under bias condition B2, the error is about 5 %. These results show that a small sensor can be designed to clearly differentiate between the effects of HCI and process variations by six months. Such a sensor can form the basis of a countermeasure against IC recycling.

While encouraging, the results presented above are only a starting point. Accurate circuit simulations, device models, and HCI degradation modelling are required for the design and validation of a feasible anti-recycling sensor. The employed simulation tools and their use is presented in the next chapter.

Chapter 4

Circuit Simulation and Data Analysis Toolchain

CHAPTER 3 laid out the case for an HCI-based anti-recycling sensor, including preliminary performance estimates showing promising results after 6 months of aging. These results offer a starting point. In order to properly carry out design and assessment of a sensor, accurate circuit simulation inclusive of HCI degradation effects is required. This chapter presents an overview of the circuit simulation tool chain and workflow used in the following chapters.

4.1 Toolchain Description

Circuit simulations were performed using HSPICE [62], the results of which were analyzed and plotted with MATLAB [63]. Conversion of HSPICE output files to MATLAB format was handled by custom Python scripts. Batch simulation runs and output processing was controlled through custom shell scripts. The computing environments was a CentOS Linux distribution run on a 24 core Intel Xeon system with 64 GB of memory.

HSPICE is Synopsys Inc.'s version of the venerable SPICE analog circuit simulator. It is widely used by the ASIC industry and is well supported by major foundries in the form

of process development kits (PDKs), which contain highly accurate device model cards. Crucially for this work HSPICE includes the MOSRA (MOSFET reliability analysis) facility, which allows for the inclusion of BTI and/or HCI effects in a circuit simulation for the purpose of predicting system lifetimes. Simulating degradation effects on a circuit is a two-step process, as depicted in Figure 4.1, which is essentially the same as was first implemented in the Berkeley BERT simulator in the early 1990s [30], [62]. The first step, the pre-stress

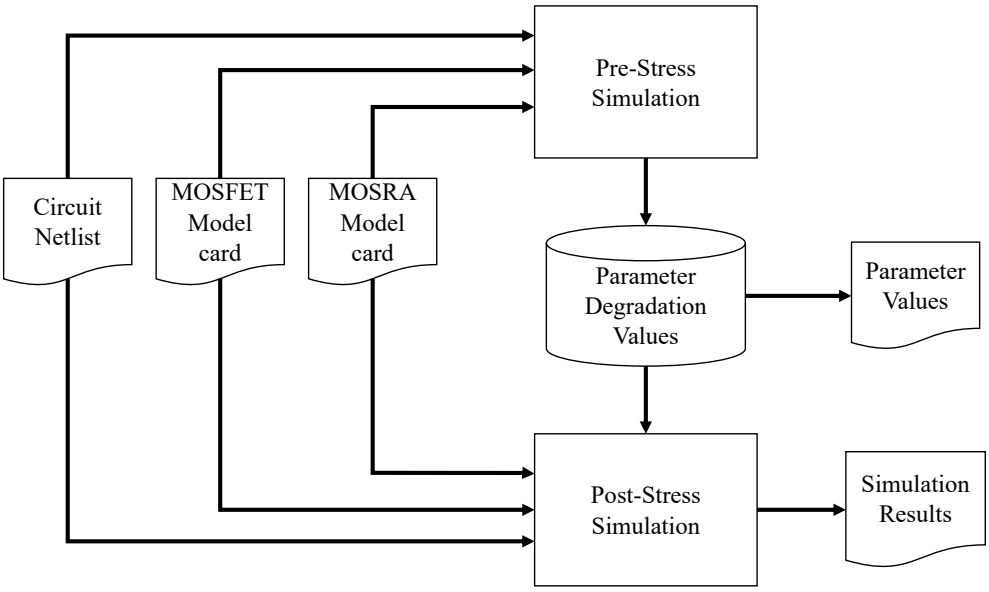


Figure 4.1: MOSRA simulation flow consists of two simulations.

simulation, determines the degradation experienced by the circuit's MOSFETs over time. The second step, the post-stress simulation, simulates the circuit using MOSFET model cards which have been modified according to the results of the pre-stress simulation. As is discussed below, splitting the simulation into two steps saves a great deal of computation and provides flexibility in circuit performance analysis.

MOSRA operates in conjunction with HSPICE circuit analyses. While the post-stress simulation may use any analysis specified by the user, the pre-stress simulation must be linked with a transient analysis. MOSRA assumes that circuit voltages are cyclical with a period of T. The transient analysis used in the pre-stress simulation must have a length of at least T. The voltage stresses on each MOSFET are determined during the transient

analysis. These stresses are then used to calculate the accumulated aging experienced by each MOSFET over T. Finally, these aging values are linearly extrapolated over time and used to determine the total degradation in the parameter of interest such as V_{TH} or I_{D}. Justification for this approach can be understood by first considering how MOSFET degradation effects are modelled. Generally, degradation models express the change in a parameter as a power law in time, taking the form

$$\Delta P(t) = C\left[f(\mathbf{V_i}) \cdot t\right]^n, \tag{4.1}$$

where t is the time, $\mathbf{V_i}$ is a set of terminal voltages, and $n \in (0, 1)$ and C are constants. These models are usually derived under DC operating conditions and cannot be directly applied to AC conditions. The difficulty arises from a violation of time invariance, i.e. the degradation is dependent on *when* voltage stress is applied. Consider a time-dependent voltage stress

$$\mathbf{V_i}(t) = \begin{cases} 0, & t < t_1 \\ \mathbf{V_i}, & t \geq t_1. \end{cases}$$

The predicted degradation $\Delta P(t_1)$ in this case is indistinguishable from the DC case where $\mathbf{V_i}$ is constant for $t \geq 0$. Intuitively degradation in the two cases should differ since there has been no stress up until t_1 in the time-varying case. Resolving this contradiction involves expressing $\frac{d}{dt}\Delta P(t)$ without an explicit time dependence and introducing a quasi-static approximation for $\mathbf{V_i}$. The solution to the resulting differential equation is

$$\Delta P(t) = C\left[\mathrm{AGE}(t)\right]^n \tag{4.2}$$

$$\mathrm{AGE}(t) = \int_0^t f\left(\mathbf{V_i}(\hat{t})\right) \cdot \hat{t}\, d\hat{t}. \tag{4.3}$$

This result means that determining $\Delta P(k \cdot T)\, \forall k$ is highly efficient since only a single integral over $[0, T]$ need be computed:

$$\begin{aligned} \Delta P(k \cdot T) &= C\left[\mathrm{AGE}(k \cdot T)\right]^n \\ &= C\left[k \cdot \mathrm{AGE}(T)\right]^n . \end{aligned}$$

At the conclusion of the pre-stress simulation the degradation values, $\Delta P(k \cdot T)$, for $k = 1 \ldots N$, are written to a file.

Different netlists may be used between the pre-stress and post-stress simulations. This allows for a different voltage stimuli and circuit analysis type to be applied in the post-stress simulation. In this way, a circuit may be operated in a different mode after some time. The post-stress simulation actually comprises of a series of N repeated circuit simulations. For each $t = k \cdot T, k = 1 \ldots N$, $\Delta P(t)$ is translated into MOSFET model parameter values which are used to update the MOSFET model card, and the circuit simulation is then run. Circuit performance in response to the degradation mechanism can thus be tracked over time.

While HSPICE provides facilities for analyzing and processing simulation results, MATLAB provides greater breadth and flexibility. This is especially true of the statistical analysis performed in chapters 5 and 6.

All circuit simulations used models from a TSMC PDK for a 65 nm process having 1 poly and 9 metal layers [64]. Though Such mature CMOS technologies, e.g., 40 nm and older, they remain relevant as they represent 54 % of the existing fab capacity [65] and continue to dominate in safety and reliability critical domains, including the automotive industry [65]. MOSFET SPICE models used are intended for RF/mixed-signal applications. They are defined as sub-circuits containing a BSIM4 compact MOSFET model and several passive components accounting for parasitics. Libraries covering typical process parameter values, four process corners (FF, SS, FS, SF), and full statistical process variations for Monte Carlo

simulations are all available. The models have been verified and are valid over a temperature range of −40 °C to 125 °C and up to 120 % of V_{DD}.

4.2 Toolchain Use

User documentation for the TSMC PDK models and MOSRA HSPICE lacks detail. This section provides some details gained through experience. Relevant workflows are described in chapters 5 and 6, and code listings are included in the appendices.

4.2.1 TSMC SPICE Models

Working with the TSMC SPICE models can appear daunting due to the many options. Thankfully TSMC has included in its PDK a set of usage libraries [66], which when included will load and properly configure all device models. Each library covers a different simulation scenario:

- `tt_lib`, `ff_lib`, `ss_lib`, `fs_lib`, or `sf_lib` each respectively cover one of the process corners TT, FF, SS, FS, or SF;
- `mc_lib` is for use with Monte Carlo simulations covering global process variations;
- `mismatch_lib` is also for Monte Carlo simulations, but for the case of local variations only.

Loading a usage library is accomplished with the following SPICE directive:

```
.lib 'PATH_TO_LIBRARY/crn65gplus_2d5_lk_v1d1_usage.l' x_lib
```

where `x_lib` is one of the above-listed libraries. MOSFETs are described by macro models using sub-circuits, which contain primitive passive models to account for parasitics and a BSIM4 MOSFET device model. A core nMOSFET macro model is imported from a library as `nmos_rf`, and a 1.8 V I/O nMOSFET as `nmos_rf_18`. The nMOSFET device within each

macro model is called `M0` and is described by the BSIM4 model `nch_rf`. Setting the geometric scale at the top-level of the netlist will not result in proper scaling of the MOSFETs because they are modelled using sub-circuits. Additional special parameters must be defined in order to do this. For core MOSFETs the parameter is `scale_mos_rf`, and for 1.8 V I/O MOSFETs it is `scale_mos_rf_18`. Both of these parameters must be defined if both devices are used in a design.

4.2.2 MOSRA

HSPICE usage is generally well documented [62]. This is unfortunately not the case when it comes to MOSRA, especially when Monte Carlo simulations are concerned. This section describes the HSPICE commands and options which must be included in a netlist for configuring and running the types of MOSRA simulations used in this work. In the following discussion, the device to be aged under HCI is a core nMOSFET. MOSRA provides three modelling approaches for degradation mechanisms. This work employs the most straightforward, the level 3 compact model.

Invoking a MOSRA simulation begins with loading and configuring a MOSRA model. It must then be attached to the MOSFET model card of the devices which are to be aged.

- The level 3 HCI model is imported as `myhcimodel`,which is an arbitrary name, and its parameter values are set as

  ```
  .model myhcimodel mosra level=3
  +HCIAP=-3.3191E-04 HCIG=2.0473E+00
  +HCID=4.8411E+00 HCIN=2.5063E-01
  ```

  Default values are automatically used for any parameters not explicitly listed.

- The MOSRA model `myhcimodel` is then appended to the nMOSFET BSIM4 model `nch_rf` through

  ```
  .appendmodel myhcimodel mosra nch_rf nmos
  ```

This must be an actual nMOSFET device model (`nch_rf`) and not the encompassing sub-circuit (`nmos_rf`). Alternatively, the MOSRA model can be attached to any and all loaded MOSFET model cards by using the wildcard `"*"` in lieu of a specific MOSFET model name.

- By default the scope of `.appendmodel` is limited to the current hierarchical level. Appending to a device model inside a sub-circuit must be explicitly allowed with

  ```
  .option appendall
  ```

- `myhcimodel` will be used to adjust `nch_rf`, but the terminal voltages of `nmos_rf` should be used to calculate the aging stress:

  ```
  .mosra_subckt_pin_volt nmos_rf
  ```

After setup of the MOSRA model, the analysis must be specified. The MOSRA analysis command has many parameters, most of which are not used in this work. The usage pattern employed in this work is

```
.mosra reltotaltime=AGESTOP relstep=AGESTEP relmode=1
+simmode=MODE_N
```

where `reltotaltime` is the desired final age of the circuit and `relstep` is the time step for reporting degradation. The degradation mechanism is selected with `relmode`, in this case it is set for HCI only. The parameter `simmode` specifies which parts of the MOSRA simulation are to be carried out:

- 0: pre-stress simulation only,
- 1: post-stress simulation only,
- 2: both pre-stress and post-stress simulations consecutively.

A regular HSPICE analysis must also be specified. This must be a transient analysis if a pre-stress MOSRA simulation will be carried out:

```
.tran TSTEP TSTOP
```

The aging stress (equation 4.3) is integrated over [0, TSTOP] and extrapolated to AGESTOP in steps of AGESTEP. The results of the pre-stress simulation will be written to a file with the extension `.radeg0`. If `simmode=2` the post-stress simulation will reuse the same transient analysis and voltage stimuli as for the pre-stress simulation. If `simmode=1` pre-stress results will be automatically loaded from a `.radeg0` file located in the working directory. A different analysis and/or voltage stimuli from the pre-stress simulation may be specified in this case. Post-stress output is controlled by HSPICE output statements, such as `.print`, and is repeated every AGESTEP until AGESTOP.

A MOSRA Monte Carlo simulation is invoked automatically by a regular Monte Carlo HSPICE analysis such as

```
.tran TSTEP TSTOP sweep monte=MCCOUNT
```

By default MOSRA does not write all results to disk for a Monte Carlo simulation. Pre-stress simulation results will no longer be saved to a file, nor will post-stress results for any age other than AGESTOP. These behaviors are controlled with the options

- `.option radegoutput=yes` will cause pre-stress results to be saved. There will be a separate file for each Monte Carlo instance.
- `.option mramcfile=1` will cause post-stress results for every AGESTEP to be saved.

There is an added complication when the pre-stress and post-stress simulations are not run consecutively (`simmode=1`). The default behavior for MOSRA in this case is to load the pre-stress results from a file in the working directory with the extension `.radeg0`. This works properly in the non Monte Carlo case. However, with Monte Carlo, there are now as many `.radeg0` files as there are Monte Carlo instances and MOSRA will only load the first of these files. The workaround is to carry out the post-stress simulation one Monte Carlo instance at a time:

1. Include `.option radegfile="MCIDX_PRESTRESS_FILENAME"` to load the pre-stress results for Monte Carlo index MCIDX.

2. Specify a single instance Monte Carlo simulation beginning at index MCIDX: `.tran TSTEP TSTOP monte sweep=1 firstrun=MCINDX`. This also works for other analysis types such as DC or AC.

A custom shell script was written to iterate over MCIDX, update the netlist file, and rerun HSPICE. This approach is possible because HSPICE does not by default use a random seed for its pseudo-random number generator, meaning that repeated Monte Carlo simulations will use the same sequence of values.

The design of the sensor presented in chapter 6 and the estimation of its performance was not possible without an accurate HCI model. The HSPICE MOSRA model, whose use is described above, had to be tuned for use with TSMC 65 nm devices. Doing so without access to a test devices nor proprietary fab data required to development of a new fitting method that exploits information in the public domain. This method is discussed in the next chapter.

Chapter 5

Leveraging Public Information to Fit a Compact Hot Carrier Injection Model to a Target Technology

CHAPTER 4 presented the operating principles and the usage of HSPICE's MOSRA facility for simulating HCI effects on TSMC devices. This was done without any discussion of the MOSRA HCI model used, which is covered in this chapter. All major electronic design automation (EDA) tools are equipped with facilities equivalent to MOSRA which simulate the effects of CMOS degradation mechanisms so that system lifetimes may be predicted. The degradation models are not generic and must be tuned to a target CMOS technology if they are to produce accurate results. Fabs are the authoritative source of degradation models and the data required to tune EDA models to their technologies, but this is sensitive information which they tightly control and do not always share with academic researchers. An alternative would be to build and characterize one's own test devices, a process which can be costly and time-consuming.

As academics engaged in early-stage design exploration of recycling countermeasures we required the ability to tune a CMOS degradation model without relying on fab data

nor the fabrication of test devices. In this chapter we present a search based method for identifying aging model parameters that leverages public domain data. Results are validated by estimating lifetime statistics through Monte Carlo simulation for comparison against a target technology's specifications.

Though we demonstrate our method with a MOSRA HCI model, it is not model specific. We focus on fitting the model to nMOSFETs from the 65 nm TSMC technology discussed in chapter 4. Fitting to pMOSFETs was not considered since they are not seriously affected by HCI [30].

We also validated our results at the circuit level by matching the accuracy of a previously reported model [67] when simulating HCI degradation in an inverter.

Our contributions through this chapter are threefold.

1. We propose a general method for fitting a MOSFET degradation model to a target technology by leveraging public domain data. The solution produces results which are statistically validated against the target technology's specifications.

2. We demonstrate the application of our method by fitting an HCI model to TSMC's 65 nm core and I/O nMOSFETs.

3. We validate our result at the circuit level.

This chapter is organized as follows. The model parameter search method, including validation, is presented in section 5.1. Section 5.2 details the public domain data used. Our model fitting results are presented in section 5.3. In section 5.4 we validate our results at the circuit level by simulating HCI aging in an inverter.

5.1 Model Parameter Search Process

We begin with a general, technology and degradation mechanism agnostic description of our process. We then discuss its use in tuning the MOSRA level 3 HCI model for use

with TSMC's 65 nm technology. There are two requirements for implementing our process. First is access to a target technology design kit, including SPICE model files and associated documentation. Second is that the degradation model of interest be of a compact parametric form, i.e. that it be a function of externally observable quantities such as terminal voltages and temperature.

Our parameter search process is iterative and consists of three steps.

1. A free search parameter, which controls the search process, is adjusted.

2. A candidate solution is determined through a least-squares fit.

3. The candidate solution is validated.

A new iteration is performed if the validation fails.

5.1.1 General Description

In order for a degradation model to accurately predict performance degradation in a target technology, the model needs to be tuned to the technology. This means identifying appropriate values for the model's parameters. This can be done by a least-squares fit if sufficient relevant data is available either directly from a fab or from measurements of test devices. Faced with a lack of such data, we developed the parameter search process depicted in Figure 5.1.

The process is initialized by gathering data for use in a least squares fit to the degradation model (Figure 5.1, step 1). We relied on two sources: the target technology's reliability rules which are a part of the technology's specifications, and public domain data sourced from peer-reviewed studies.

The reliability rules describe the device time to failure (TTF) statistical distribution under an operating condition selected to accentuate a given degradation mechanism. The specified operating condition and the TTF information are used to create a synthetic data

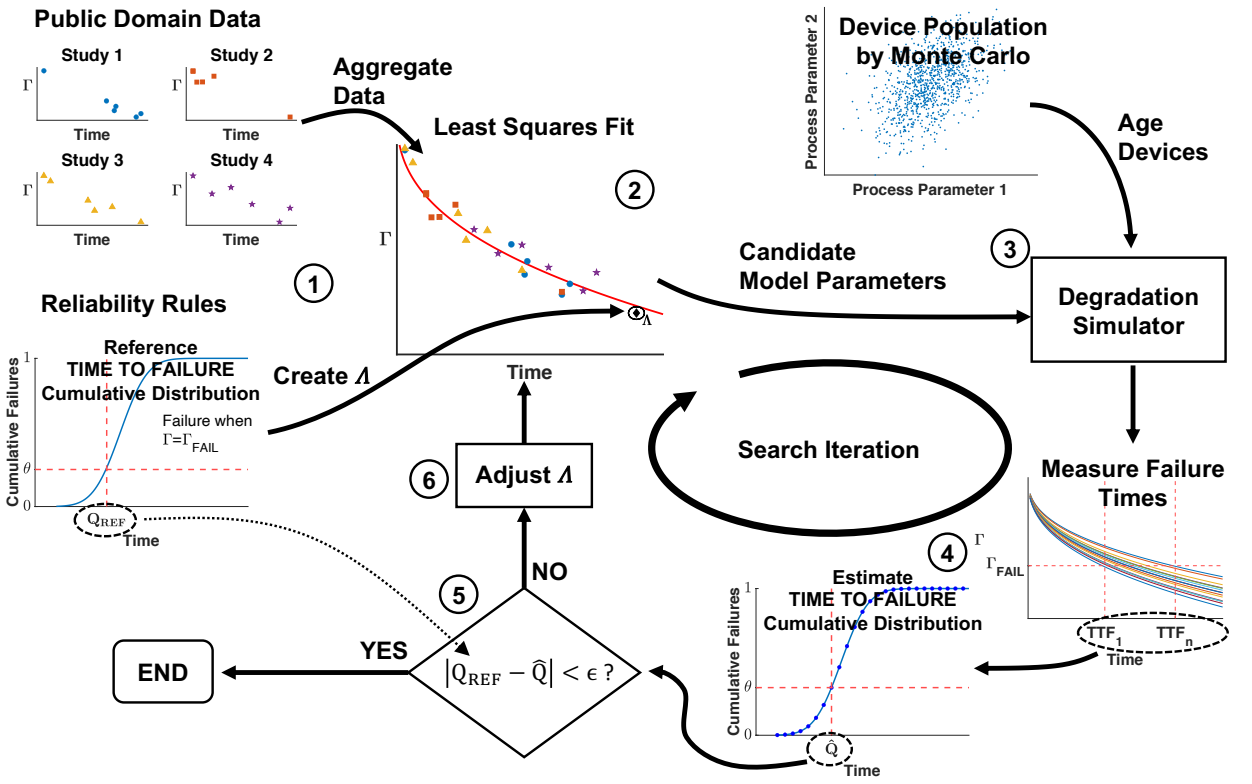


Figure 5.1: Illustration of the degradation model parameter search process. The degradation mechanism causes a reduction in the quantity Γ over time. 1. Data is aggregated from the public domain and Λ is constructed from the target technology reliability rules. In this illustration the reliability rules specify the TTF quantile Q_{REF}, the time by which a portion θ of devices fail. 2. A least squares fit is performed to determine candidate model parameters. 3. The candidate model parameters are used to simulate aging in a population of devices generated through a Monte Carlo process. 4. A time to failure distribution is computed from the simulation results and used to estimate $\hat{Q}$, the time by which θ of the population failed. 5. The validation checks the estimate $\hat{Q}$ against the reference Q_{REF}. 6. If the validation fails Λ is adjusted and a new iteration is performed.

point Λ which serves as the free search parameter. As will be explained shortly, Λ introduces an extra degree of freedom into the fitting process.

This synthetic point is intended to be a plausible degradation measurement of a target technology device. If the reliability rules provide the average TTF, Λ would represent a measurement from a typical device. If the average is not provided, other available TTF information such as a distribution quantile must be used. In such a case, Λ would represent a measurement from a realistic, though atypical device. Only one such point can be constructed

because the reliability rules do not specify multiple operating conditions. Being a single data point, Λ cannot be used on its own to perform a least-squares fit.

Additional data which covers multiple operating conditions spanning the degradation model's dimensions is required to perform a least-squares fit. This was sourced from publicly available peer-reviewed studies. Unfortunately a single study is unlikely to provide sufficient data to span all of a degradation model's dimensions. Thus we aggregated data from multiple studies. A complicating factor is that few studies explicitly identify the precise CMOS technology involved. In response we made the assumption that the effects of a degradation mechanism generalize across similar CMOS technologies. Since materials and device geometries are similar across such technologies, we surmised that degradation effects are also similar. Under this assumption, degradation model parameters for similar technologies fall into a near-neighborhood of the parameter space. Model parameters found through a least-squares fit to data aggregated from several similar technologies will also fall into this near-neighborhood. By iteratively adjusting Λ this near-neighborhood is searched until a solution for a specific target technology can be validated.

A least squares fit to the aggregated data is performed to determine a set of candidate degradation model parameters (Figure 5.1, step 2). Because most or even all of the public domain data may not be from the target technology, the role of Λ is to bias the fit towards the target technology. The parameter near-neighborhood is searched by adjusting Λ. Different adjustments are possible depending on the information provided by the reliability rules. One possibility may be to change Λ's weight relative to the rest of the data. This strategy may be a good choice if the average TTF is used in constructing Λ. If only a TTF quantile is available, adjusting the time value of Λ may be an option. In this way, Λ may be moved towards a more typical measurement.

Validation of a candidate parameter set consists of generating estimated lifetime statistics for comparison against the reliability rules. A population of devices is first generated in a Monte Carlo simulation using a process, voltage, and temperature (PVT) variation device

model provided by the target technology's design kit. This population is then aged according to the degradation model using the candidate parameter set (Figure 5.1, step 3) and the same operating conditions as in the reliability rules. An estimate of the TTF distribution is then calculated from this simulation data (Figure 5.1, step 4) and compared to the reference distribution from the reliability rules (Figure 5.1, step 5). If the validation fails, a new search iteration begins with an adjustment to Λ (Figure 5.1, step 6).

In summary, our model parameter search process is initialized with the construction of Λ from the target technology's reliability rules, and the collection of relevant public domain data. The iterative search begins with the adjustment of the free search parameter. This may be the weight of Λ and/or Λ's degradation time. Next, a least-squares fit of the degradation model to the public domain data along with Λ is performed. Finally, the candidate model parameters are validated by their use in aging a simulated population of devices for comparison against the reliability rules. A new iteration is begun if the validation fails.

5.1.2 Application Example

We now describe the application of our parameter search process to fitting an HCI model from Synopsys HSPICE [62] for use with TSMC's 65 nm technology. We first describe the HCI model as well as pertinent information from the reliability rules. We then describe the construction of Λ, including its initialization. Finally we describe the iterative search steps.

HSPICE is used to generate and age the simulated device population. Results are then pre-processed with a custom Python script and imported into MATLAB[63] where the rest of the steps are performed. Figure 5.2 depicts the implementation.

5.1.2.1 HCI Model

MOSRA provides three modeling approaches for HCI. The most straightforward to work with is the level 3 model which is a parameterized compact model, which expresses degradation

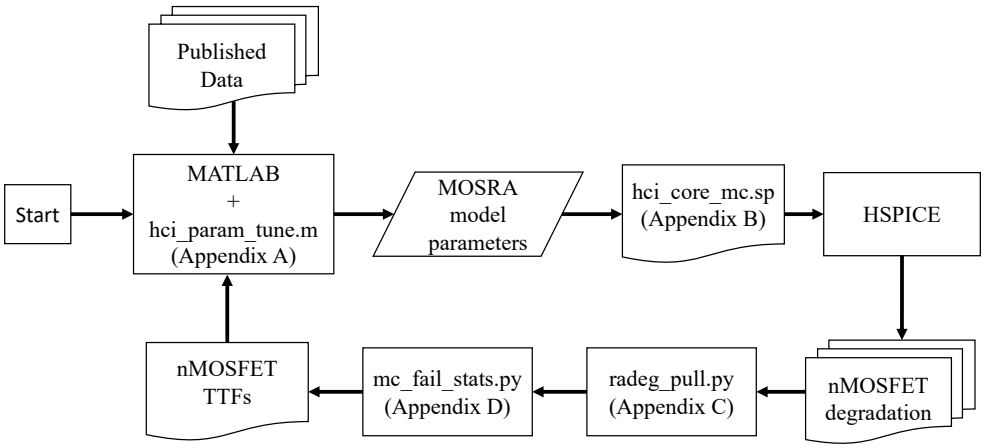


Figure 5.2: Implementation of the fitting method for the MOSRA level 3 HCI model. Code listings are included in the appendices noted in the figure.

as a percentage change in the drain current:

$$\Delta I_D(\%) = A_{\mathrm{p}} \cdot \mathrm{e}^{(G \cdot V_{\mathrm{GS}})} \cdot \mathrm{e}^{(D \cdot V_{\mathrm{DS}})} \cdot \mathrm{e}^{\left(\frac{-E_{\mathrm{a}}}{kT}\right)} \cdot \mathrm{e}^{(-m \cdot L)} \cdot t^n. \tag{5.1}$$

The stresses giving rise to this degradation are described by the variables listed in Table 5.1. It should be noted that the model depends on the polysilicon length (i.e. gate length) rather than the actual channel length. Table 5.2 lists the model's parameters, the values of which are target technology dependent. Synopsys suggests default values for several parameters which are claimed to be sufficient for many situations [62]. For brevity, (5.1) will be referred to as the MOSRA model in the rest of this work.

Table 5.1: MOSRA HCI Compact Model Variables [62].

Variable	Unit	Description
V_{GS}	V	Gate-source bias.
V_{DS}	V	Drain-source bias.
T	K	Channel temperature.
L	µm	Polysilicon length.
t	s	Total stress time.

Table 5.2: MOSRA HCI Compact Model Parameters [62].

Parameter	Unit	Default Value (Synopsys)	Description
A_p	-	0	Pre-factor.
G	V^{-1}	0	V_GS dependence.
D	V^{-1}	9.0	V_DS dependence.
E_a	eV	0.05	Temperature acceleration.
m	$\mu\mathrm{m}^{-1}$	2.0	Polysilicon length dependence.
n	-	0.5	Time dependence.

When describing a set of measurements, the MOSRA model can be logarithmically scaled and expressed in matrix form as

$$\mathbf{\Delta I_{D,log}} = \mathbf{S} \cdot \mathbf{v}, \tag{5.2}$$

where the vectorized MOSRA model variables are grouped as the matrix

$$\mathbf{S} = \begin{bmatrix} \mathbf{1} & \mathbf{V_{GS}} & \mathbf{V_{DS}} & (k\mathbf{T})^{-1} & \mathbf{L} & \ln(\mathbf{t}) \end{bmatrix}$$

and the MOSRA model parameters form the vector

$$\mathbf{v} = \begin{bmatrix} \ln(A_\mathrm{p}) & G & D & -E_\mathrm{a} & -m & n \end{bmatrix}^\mathrm{T}.$$

For d data points, $\mathbf{\Delta I_{D,log}} \in \mathbb{R}^d$, $\mathbf{S} \in \mathbb{R}^{d\times 6}$, and $\mathbf{v} \in \mathbb{R}^6$. This system has a unique least squares solution $\mathbf{v}$ while the matrix $\mathbf{S}$ is full column rank [68].

5.1.2.2 Reliability Rules

The TSMC 65 nm reliability rules pertaining to HCI [58] define a failure as a 10 % reduction in drain current in the saturation region ($I_\mathrm{D,sat}$) relative to initial conditions. For each device type (e.g. core and I/O), a constant voltage and temperature operating condition selected to accentuate HCI stress are prescribed and the resulting 10^{-3} TTF quantile, $Q_\mathrm{REF}^{0.001}$, is given. This is the time by which one out of a thousand devices is expected to have failed under

the prescribed operating conditions. Validation consists of comparing $Q_{\mathrm{REF}}^{0.001}$ to $\hat{Q}^{0.001}$, the estimated the 10^{-3} TTF quantile which results from a candidate set of model parameters $\mathbf{v}$.

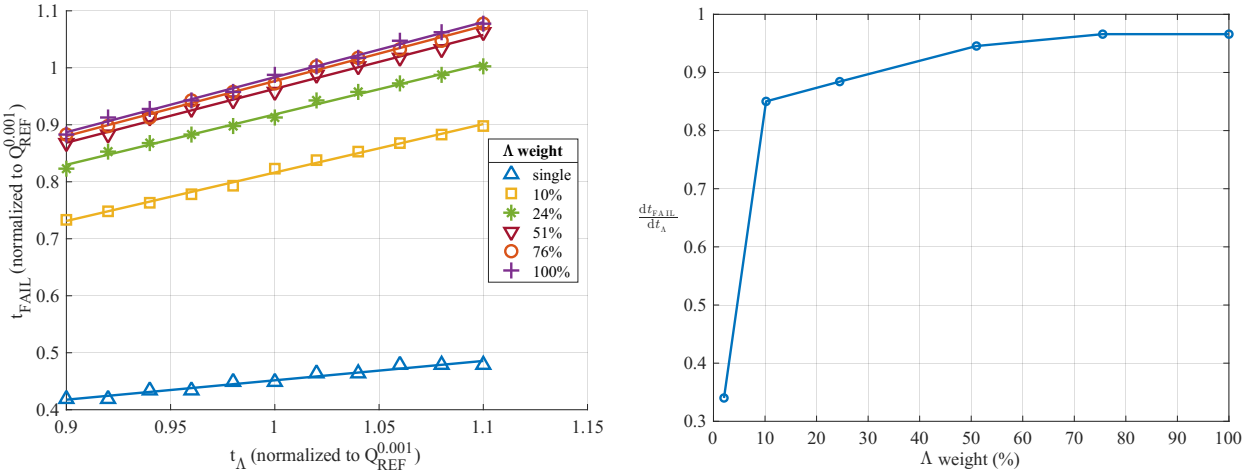
5.1.2.3 Free Search Parameter Initialization

Λ, which corresponds to a row in $[\mathbf{\Delta I_{D,log}}|\mathbf{S}]$, is constructed from the reliability rules. Its $\Delta I_{\mathrm{D,log}}$ is set to $\ln(10)$ since a failure is a 10 % degradation. The variables $V_{\mathrm{GS}}, V_{\mathrm{DS}}, T,$ and L are as specified in the reliability rules.

Because the reliability rules only provide $Q_{\mathrm{REF}}^{0.001}$, we use the degradation time t_Λ as the free search parameter. It is initialized to $Q_{\mathrm{REF}}^{0.001}$ and adjusted in each search iteration. Adjustments to t_Λ are informed by the mapping $t_\Lambda \rightarrow \hat{Q}^{0.001}$, the exact form of which is unknown. However, so long as the adjustments are kept small, the mapping can be approximated as linear and calculated iteratively.

Setting the weight of Λ relative to the rest of the data during fitting is done at this point. A desirable weight for Λ should allow $\hat{Q}^{0.001}$ to quickly reach $Q_{\mathrm{REF}}^{0.001}$ with a few small adjustments to t_Λ. As its weight increases, Λ becomes more influential in the fitting steps. This causes $\hat{Q}^{0.001}$ to be more sensitive to t_Λ. Too large a weight however risks causing Λ to completely overshadow the rest of the data. To determine an appropriate weight value, the dependence of $t_\Lambda \rightarrow \hat{Q}^{0.001}$ on Λ's weight was estimated for several weight values. To avoid performing the great number of Monte Carlo simulations which would be required to calculate $\hat{Q}^{0.001}$, $t_\Lambda \rightarrow \hat{Q}^{0.001}$ was approximated as $t_\Lambda \rightarrow t_{\mathrm{FAIL}}$, where t_{FAIL} is the time at which the MOSRA model predicts a 10 % degradation in a typical device under the reliability rules' operating conditions. Effectively, t_{FAIL} is the mean TTF resulting from a given $\mathbf{v}$. We assumed that the TTF distribution had a small variance, resulting in $\hat{Q}^{0.001}$ to be close to t_{FAIL}. Figure 5.3 shows the result of sweeping t_Λ by ± 10 % from its initial value for various weights of Λ. The weight was adjusted by adding extra copies of Λ to the fitting data. A weight of 100 % corresponds to as many copies as there are public domain data points.

Here a weight of 50 % is a good choice as this allows t_{FAIL} to reach $Q_{\text{REF}}^{0.001}$ with only a small adjustment to t_Λ. Increasing the weight past this does not offer any significant advantage.



(a) **(b)**

Figure 5.3: (a)Dependence of t_{FAIL} on t_Λ for various weights of Λ relative to the number of public domain data points. A weight of 100 % means Λ has as much weight as all public domain points combined. A weight of single means Λ has a weight equal to a single public domain point. (b) The sensitivity of t_{FAIL} to t_Λ has a dependence on the weight of Λ. Increasing the weight beyond a certain point, about 50 % here, yields little benefit.

5.1.2.4 Free Search Parameter Adjustment

Adjusting t_Λ depends on the current search iteration. In the i^{th} iteration t_Λ and $\hat{Q}^{0.001}$ are labelled as $t_{\Lambda,i}$ and $\hat{Q}_i^{0.001}$ respectively. We also introduce $\Delta\hat{Q}_{i-1}^{0.001} = Q_{\text{REF}}^{0.001} - \hat{Q}_{i-1}^{0.001}$.

1. For $i = 1 : t_{\Lambda,i} = Q_{\text{REF}}^{0.001}$.
2. For $i = 2$: $t_{\Lambda,i} = t_{\Lambda,i-1} + \frac{\partial t_\Lambda}{\partial t_{\text{FAIL}}}\Delta\hat{Q}_{i-1}^{0.001}$.
3. For $i > 2$: $t_{\Lambda,i} = t_{\Lambda,i-1} + \frac{\partial t_\Lambda}{\partial \hat{Q}^{0.001}}\Delta\hat{Q}_{i-1}^{0.001}$.

$\frac{\partial t_\Lambda}{\partial t_{\text{FAIL}}}$ was found when determining Λ's weight (see Figure 5.3). $\frac{\partial t_\Lambda}{\partial \hat{Q}^{0.001}}$ is derived from a linear regression to the set of pairs $\left\{\left\langle t_{\Lambda,j}, \hat{Q}_j^{0.001}\right\rangle \mid j = 1 \ldots i\right\}$. A new $\mathbf{v}$ is computed based on the updated value of t_Λ.

5.1.2.5 Least Squares Fit

S, which is composed of public domain data and multiple copies of Λ, remains largely unchanged between search iterations. Only t_Λ is updated. Once this is done, a new least squares solution **v** to (5.2) is computed.

5.1.2.6 Validation

If $\left|\hat{Q}^{0.001} - Q^{0.001}_{\mathrm{REF}}\right| > \epsilon$ the validation fails and a new search iteration begins. To compute $\hat{Q}^{0.001}$ we need a TTF sample population. One of the global variation nMOSFET models described in chapter 4 is used to generate a sample population of devices which are then individually aged using the MOSRA model with the current value of **v**. When estimating a quantile Q^θ from a Monte Carlo simulation, the sample population should be larger than θ^{-1} [56], hence to estimate the 10^{-3} quantile we used a sample population of 10^4. The voltages and temperature are those specified by the reliability rules. This is all carried out in HSPICE. We then compute an estimate of the TTF cumulative distribution function (CDF) with the Kaplan-Meier estimator, a nonparametric estimator often used in failure analysis that can be shown to maximize the likelihood over the space of all distributions [69]. Once the CDF is estimated, $\hat{Q}^{0.001}$ is found by a simple lookup operation. We also calculate the confidence bounds on $\hat{Q}^{0.001}$ by estimating the confidence bounds on the TTF CDF using a Gaussian distribution. This is possible because the standard large sample theory for maximum likelihood estimators applies [69]. The variance used in this estimate is given by Greenwood's formula [69].

5.2 Data Sources

The public domain data needed to satisfy three criteria:

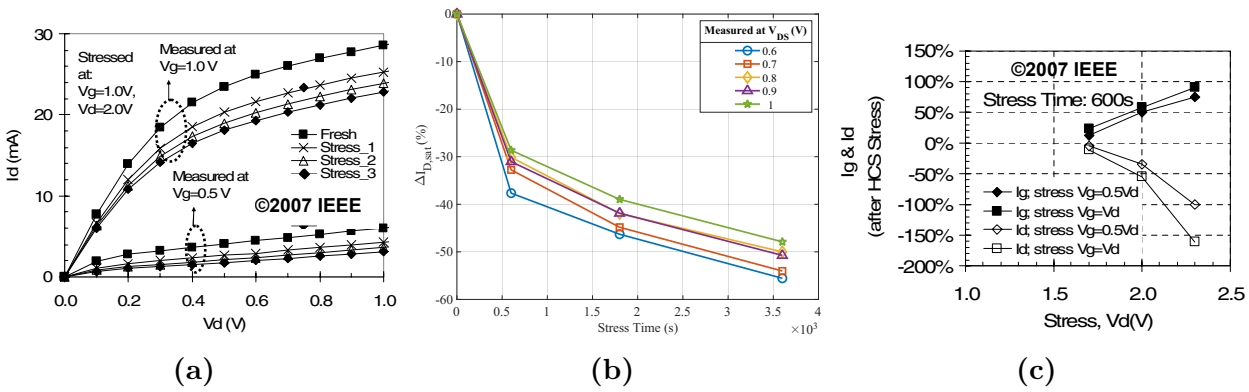
1. express HCI degradation as a decrease in $I_{\mathrm{D,sat}}$;

2. have been measured from devices from a 65 nm technology;

3. fully span **S**, meaning that the recorded HCI degradation was in response to variations in all of the dimensions of **S**.

Identifying peer-reviewed studies which met these criteria was challenging. To increase the pool of data we assumed that the MOSRA model generalizes across technologies of comparable node size. This allowed us to combine data from different studies so long as they recorded HCI degradation as a shift in drain current in a 65 nm technology device. Even so, we were unable to identify enough data to entirely span **S**. To resolve this we employed the default values suggested by Synopsys for the parameters E_a (temperature acceleration) and m (polysilicon length dependence) (see Table 5.2), reducing the dimensionality of the problem by two. We identified the 5 studies described below which met our needs. These studies are not all recent since 65 nm technologies are now several generations old, although still in common use. We used WebPlotDigitizer [70] to extract data from the plots reproduced below.

Figure 5.4 is drawn from Fakhruddin et al., which examined the RF performance degradation of nMOSFETs from an unspecified 65 nm technology with test devices whose dimensions were $W/L = 1.2\,\mu\mathrm{m}/0.06\,\mu\mathrm{m}$ [71]. Figure 5.4(a) and Figure 5.4(c) reproduce the relevant HCI degradation measurements. All stress voltages were DC and are specified on the plots. Figure 5.4(a) shows degradation in response to constant voltage stress as a series of $I_\mathrm{D} - V_\mathrm{DS}$ scans performed over time. We used a subset of the depicted measurements. We used data measured under $V_\mathrm{GS} = 0.5\,\mathrm{V}$ since these values are consistent with Figure 5.4(c). Of these values, we only wanted $I_\mathrm{D,sat}$ measurements so we limited the V_DS measurement range to $\geq 0.6\,\mathrm{V}$. Figure 5.4(b) shows our remapping of this data to relative $I_\mathrm{D,sat}$ degradation. Figure 5.4(c) shows the relative change in $I_\mathrm{D,sat}$ after being subjected to various DC voltage stresses for 600 s. It should be noted that the degradation for a V_DS stress of 2.3 V was $\Delta I_\mathrm{D,sat} \leq -100\,\%$, which is physically meaningless. These points were treated as outliers and excluded. The temperature was not reported and assumed to be 25 °C.



(a) **(b)** **(c)**

Figure 5.4: HCI measurements from Fakhruddin et al. [71]. (a) $I_D - V_{DS}$ scans performed after 0 s (Fresh), 600 s (Stress_1), 1800 s (Stress_2), and 3600 s (Stress_3) of constant $V_{GS} = 1\,V$ and $V_{DS} = 2\,V$ stress [71]. (b) Our conversion of (a) to relative $I_{D,sat}$ degradation over time. (c) Relative $I_{D,sat}$ degradation after 600 s of various DC voltage stresses (open symbols) [71].

Figure 5.5 is drawn from Yuan et al., which reported on the performance degradation of a low noise amplifier when subjected to HCI stress [72]. This work was carried out using minimum size devices in TSMC's 65 nm technology. nMOSFET I_D degradation over time was measured as the six $I_D - V_{DS}$ scans reproduced in Figure 5.5. The applied V_{DS} stress

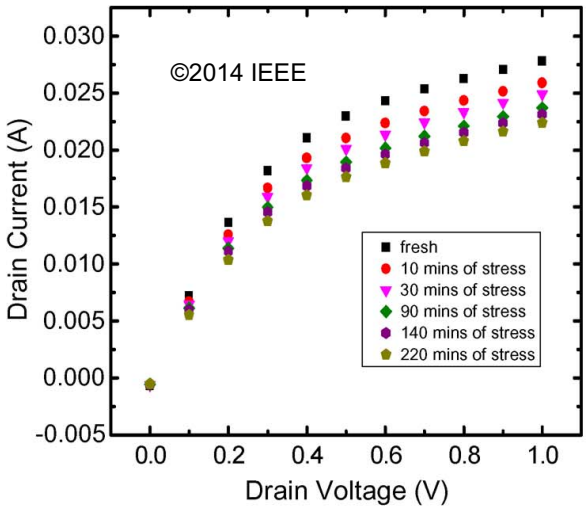
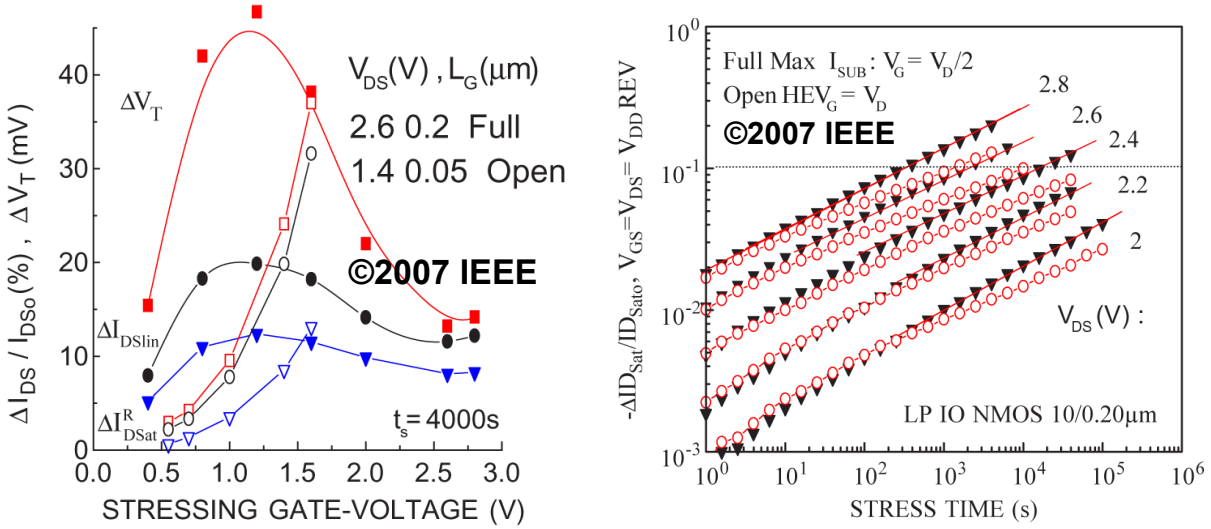


Figure 5.5: HCI measurements from Yuan et al. [72].

was 2 V and the over voltage (V_{OV}) stress was 0.35 V. We were able to calculate the stress V_{GS} from the provided V_{OV} by making use of the threshold voltage, which we knew from the design kit [58]. To limit the data to $I_{D,sat}$ we excluded points scanned at $V_{DS} < 0.6\,V$. The

temperature was not reported and assumed to be 25 °C. We converted these measurements to relative $I_{\mathrm{D,sat}}$ degradation over time using the same process which yielded Figure 5.4(b).

Figure 5.6 is drawn from Huard et al. [67]. This work describes a quantitative assessment methodology for HCI and NBTI degradation effects in an unspecified technology of a node size in the 90 nm to 65 nm range. Figure 5.6(a) shows HCI dependence on stress V_{GS} for



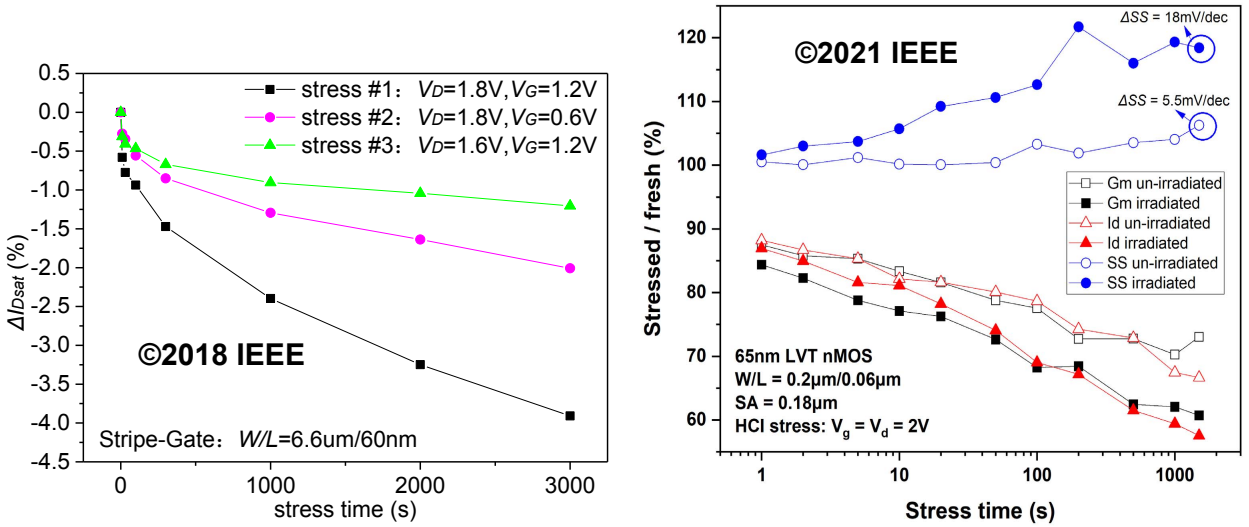
(a) **(b)**

Figure 5.6: HCI measurements from both core and I/O nMOSFETs [67]. (a) HCI dependence on stress V_{GS} for core (empty triangles) and I/O (filled triangles) nMOSFETs. (b) HCI degradation in an I/O nMOSFET when subjected to V_{DS} ranging from 2 V to 2.8 V and V_{GS} equal to either V_{DS} (circles) or $\frac{1}{2}V_{\mathrm{DS}}$ (triangles).

both core (empty triangles) and I/O nMOSFETs (filled triangles). The MOSRA model, which has an exponential dependence on V_{GS}, is a poor match for the I/O device data. Two workarounds are possible. First, we can assume that HCI degradation is independent of V_{GS} in I/O devices. This may be justified by observing that the data suggests HCI dependence on V_{GS} is far weaker in I/O devices than in core devices. Second, we can limit the V_{GS} range to values larger than $\frac{1}{2}V_{\mathrm{DS}}$. The data suggests that HCI degradation peaks when $V_{\mathrm{GS}} \approx \frac{1}{2}V_{\mathrm{DS}}$, which is expected in longer channel devices [30]. This would limit the applicable voltage range for the MOSRA model to $V_{\mathrm{GS}} \geq \frac{1}{2}V_{\mathrm{DS}}$. Figure 5.6(b) shows HCI degradation in an I/O nMOSFET with $W/L = 10\,\mu\mathrm{m}/0.2\,\mu\mathrm{m}$ as a change in $I_{\mathrm{D,sat}}$ relative magnitude over time

when subjected to various voltage stresses. The temperature was assumed to be 25 °C in all cases.

Figure 5.7(a) and Figure 5.7(b) are drawn respectively from Xie et al. [73] and Ren et al. [74]. Figure 5.7(a) shows relative $I_{\mathrm{D,sat}}$ degradations over time under the three bias conditions listed in the figure for a conventional planar geometry nMOSFET with $W/L = 6.6\,\mu\mathrm{m}/0.06\,\mu\mathrm{m}$ fabricated in an unspecified 65 nm technology. These measurements formed part of a study of HCI in annular gate devices intended for radiation hardened circuits [73]. Figure 5.7(b) depicts HCI degradation in irradiated and unirradiated devices in response to the bias condition described on the figure [74]. The test devices had a $W/L = 0.2\,\mu\mathrm{m}/0.06\,\mu\mathrm{m}$ and were fabricated in an unspecified 65 nm technology. We only used measurements from unirradiated devices. The temperature was reported for neither Figure 5.7(a) nor Figure 5.7(b) and was assumed to be 25 °C in both cases.



(a) **(b)**

Figure 5.7: (a) HCI measurements from Xie et al. [73]. (b) HCI measurements from Ren et al. [74]. Only I_{D} measurements from the unirradiated device (open triangles) were used.

5.3 Model Fitting Results

We obtained a total of five sets of MOSRA model parameters for TSMC 65 nm standard threshold voltage (STDVT) nMOSFETs: three for core devices and two for I/O devices. The different parameter sets were obtained using different subsets of the public domain data as detailed in Table 5.3.

Table 5.3: Public domain data used for determining the parameter sets. $\mathbf{v_{core}}$ are for core nMOSFETs and $\mathbf{v_{IO}}$ are for I/O nMOSFETs.

	Parameter Set				
	$\mathbf{v_{core1}}$	$\mathbf{v_{core2}}$	$\mathbf{v_{core3}}$	$\mathbf{v_{IO1}}$	$\mathbf{v_{IO2}}$
Figure 5.4	✓		✓		
Figure 5.5	✓		✓		
Figure 5.6(a)	✓		✓		✓
Figure 5.6(b)				✓	✓
Figure 5.7(a)		✓	✓		
Figure 5.7(b)		✓	✓		

The parameters sets for I/O devices resulted from two approaches to using the I/O-relevant data in Figure 5.6(a), which cannot be easily modelled by an exponential function. $\mathbf{v_{IO1}}$ resulted from assuming HCI degradation is independent of V_{GS}. It was obtained without the use of Figure 5.6(a). $\mathbf{v_{IO2}}$, which assumes dependence on V_{GS}, was obtained by incorporating Figure 5.6(a) data for which $V_{\mathrm{GS}} \geq \frac{1}{2}V_{\mathrm{DS}}$. Consequently $\mathbf{v_{IO2}}$ is only valid over this voltage range.

The parameter sets for core devices were obtained from three groupings of the public domain data. We split the data into two groups based on their geographic origins: a Western group ($\mathbf{v_{core1}}$) and a Chinese group ($\mathbf{v_{core2}}$). These two groups were amalgamated into a third group ($\mathbf{v_{core3}}$). This was done to judge our underlying assumption that HCI effects are similar in similar technologies. Under the same conditions, devices from different but similar technologies should exhibit similar degradations. To examine this, we simulated the public domain data using a TSMC model and the different MOSRA model parameter sets. If all three $\mathbf{v_{core}}$ sets were to result in comparable simulation errors, then HCI effects would be

virtually technology-independent. Otherwise process parameters other than node size would play a role.

Table 5.4 lists the values for our different parameter sets, and Table 5.5 summarizes the search settings. The number of data points is for the combined number of public domain points used. Λ weight is defined relative to this number. A weight of w relative to d data points means that wd copies of Λ were used. The search process was fast, reaching convergence in no more than three iterations.

Table 5.4: Derived MOSRA HCI parameter values. Parameters E_a and m assume their default values (see Table 5.2) in all cases.

Parameter Set	A_p	G	D	n
$\mathbf{v_{core1}}$	3.3191×10^{-4}	2.0473	4.8411	0.25063
$\mathbf{v_{core2}}$	1.1271×10^{-4}	2.2461	4.1104	0.33726
$\mathbf{v_{core3}}$	6.0821×10^{-6}	1.5335	6.3221	0.40415
$\mathbf{v_{IO1}}$	1.4171×10^{-3}	0	3.5089	0.25805
$\mathbf{v_{IO2}}$	1.3961×10^{-3}	-5.6519×10^{-2}	3.5624	0.25576

Table 5.5: MOSRA model parameter search process settings.

	Parameter Set				
	$\mathbf{v_{core1}}$	$\mathbf{v_{core2}}$	$\mathbf{v_{core3}}$	$\mathbf{v_{IO1}}$	$\mathbf{v_{IO2}}$
Data Points	49	32	81	225	229
Λ Weight	0.5	0.4	0.4	1.0	0.5
Monte Carlo Time Step (years)	5×10^{-3}	5×10^{-3}	5×10^{-3}	5×10^{-4}	5×10^{-4}
Search Iterations	2	2	3	3	3

All of our parameter sets passed the validation test with a high degree of confidence, as summarized in Table 5.6. $Q_{REF}^{0.001}$ falls within the 99 % confidence bounds on $\hat{Q}^{0.001}$ in all cases. From the sizes of the confidence bounds, we can conclude that $\hat{Q}^{0.001}$ differs from $Q_{REF}^{0.001}$ by at most 0.2 % (5.5 days) 99 times out of 100 for $\mathbf{v_{core}}$, and by at most 0.5 % (1.2 days) 99 times out of 100 for $\mathbf{v_{IO}}$.

Table 5.7 summarizes the root mean square (RMS) relative errors of the public domain data simulations using TSMC device models. The average error for $\mathbf{v_{core1}}$ taken over the data used in its derivation is 34 %. The equivalent figure for $\mathbf{v_{core2}}$ is 40 %. These results

Table 5.6: MOSRA model parameter validation. All values have units of years.

Parameter Set	$Q_{\mathrm{REF}}^{0.001}$	$\hat{Q}^{0.001}$	99 % Confidence Bounds	
			Lower	Upper
$\mathbf{v_{core1}}$	6.75	6.755	6.750	6.760
$\mathbf{v_{core2}}$	6.75	6.750	6.745	6.765
$\mathbf{v_{core3}}$	6.75	6.750	6.750	6.755
$\mathbf{v_{IO1}}$	0.6663	0.666	0.6635	0.6695
$\mathbf{v_{IO2}}$	0.6663	0.666	0.663	0.6695

suggest that $\mathbf{v_{core1}}$ and $\mathbf{v_{core2}}$ represent technology groupings which are comparably related to the target technology. This seems to support our assumption that HCI effects depend on little more than the node size. However, the average error for $\mathbf{v_{core3}}$ is much larger, 63 %. This apparent contradiction can be clarified by noticing that $\mathbf{v_{core1}}$ and $\mathbf{v_{core2}}$ both performed very poorly when simulating data used in each other's derivation. This suggests that $\mathbf{v_{core1}}$ and $\mathbf{v_{core2}}$ represent mutually exclusive technology groupings; consequently, HCI effects depend upon process parameters in addition to node size. Care must therefore be taken when selecting appropriate data. The average error results suggest that the $\mathbf{v_{core1}}$ parameters are more closely related to the target technology than the $\mathbf{v_{core2}}$ parameters. Hence we favor the $\mathbf{v_{core1}}$ parameter set. The $\mathbf{v_{IO}}$ parameter sets led to near identical simulation errors and cannot be clearly differentiated.

Table 5.7: RMS relative error for data set simulations using the MOSRA model and TSMC 65 nm device models.

	Parameter Set				
	$\mathbf{v_{core1}}$	$\mathbf{v_{core2}}$	$\mathbf{v_{core3}}$	$\mathbf{v_{IO1}}$	$\mathbf{v_{IO2}}$
Figure 5.4	32 %	82 %	49 %	N/A	N/A
Figure 5.5	18 %	80 %	12 %	N/A	N/A
Figure 5.6(a)	50 %	73 %	72 %	N/A	15 %
Figure 5.6(b)	N/A	N/A	N/A	19 %	20 %
Figure 5.7(a)	510 %	38 %	100 %	N/A	N/A
Figure 5.7(b)	267 %	42 %	82 %	N/A	N/A

5.4 Circuit Level Validation

Ultimately, our interest lay in using the MOSRA model to predict circuit-level performance degradation. We therefore wanted to validate our derived parameter sets against circuit-level measurements. Ideally, we wanted to find a published report of HCI degradation in a simple circuit like an inverter. Fortunately, Huard et al. provided just such information [67]. They reported on a custom HCI model's ability to replicate the HCI degradation measured in an inverter's nMOSFET. They accelerated HCI degradation by operating their inverter at an elevated temperature of 125 °C and driving it with specialized pulses. These pulses, specified in Table 5.8, had long rise and fall times in order to lengthen the time during which the nMOSFET conducted current. For each pulse type a single physical measurement and a single simulation result were reported without a corresponding age. These are reproduced in Table 5.9. For discussion clarity, the set of their simulation results will be referred as S_{H}, the set of their physical measurements as D_{H}, and their model as M_{H}.

Table 5.8: Inverter stimuli [67].

Pulse Type	f (MHz)	t_{r} (µs)	t_{f} (µs)	Amplitude (V)
B	0.1	1.0	1.0	$V_{\mathrm{DD}}/4$
C	0.1	1.0	1.0	$V_{\mathrm{DD}}/2$
D	1.0	0.1	0.1	$V_{\mathrm{DD}}/2$

Table 5.9: Reported inverter nMOSFET $I_{\mathrm{D,sat}}$ percentage degradation from measurement (D_{H}) and simulation (S_{H}) [67].

Pulse Type	D_{H}	S_{H}
B	-1.4	-2.3
C	-0.8	-0.9
D	-1.6	-1.0

We could not directly compare our own simulation results against D_{H} since the degradation values were reported without the corresponding ages. However, through educated guessing we were able to test whether we could match the accuracy of the M_{H} model.

We began by assuming that the different reported inverter degradations were all measured at the same age. We then ran our own simulations and tested whether our results all occurred within overlapping windows of uncertainty. The windows represent bounds on the accuracy of our model. The upper bound being perfect accuracy and the lower being equal to the accuracy of the M_{H} model. Figure 5.8 illustrates the principle. Each curve corresponds to our own simulation results of the inverter driven by a different pulse. Onto each curve we map the physical measurement (D_{H}) for the same pulse type. Each point has vertical error bars equal to $|D_{\mathrm{H}} - S_{\mathrm{H}}|$ representing the accuracy of the M_{H} model. These vertical error bars are then mapped onto our curves to create a corresponding error in time. If the time error bars overlap, then the lower bound on the accuracy of our simulations is equal to the accuracy of the M_{H} model. The upper bound is perfect replication and coincides with the D_{H} points aligning in time.

In addition to the aging time, the device dimensions, inverter load, and V_{DD} were unspecified. We assumed minimum channel length core devices were used and set $W/L = 1\,\mu\mathrm{m}/0.06\,\mu\mathrm{m}$ for our nMOSFET and $W/L = 2\,\mu\mathrm{m}/0.06\,\mu\mathrm{m}$ for our pMOSFET. We added two inverters in series to act as a load. These were each scaled $10\times$ larger than the driving inverter. We considered three candidate values for V_{DD}: 1.0 V, the nominal voltage for TSMC 65 nm core devices; 1.1 V, their maximum rated voltage; and 1.5 V, which was used elsewhere by Huard et al. [67]. We used 1.5 V since it was the only V_{DD} value which would lead to the required degradation within one year for all input pulses.

Our simulations were carried out in HSPICE using typical corner device models from the TSMC 65 nm design kit. HCI degradation was simulated in all nMOSFETs using the MOSRA model with parameter set $\mathbf{v_{core1}}$. Figure 5.9 shows the degradation of our driving inverter's nMOSFET along with the D_{H} points. The error bars in time all overlap, confirming that the accuracy of our HCI model is bounded by that of the previously-reported M_{H} model. Doubling the size of the load inverters did not change this result.

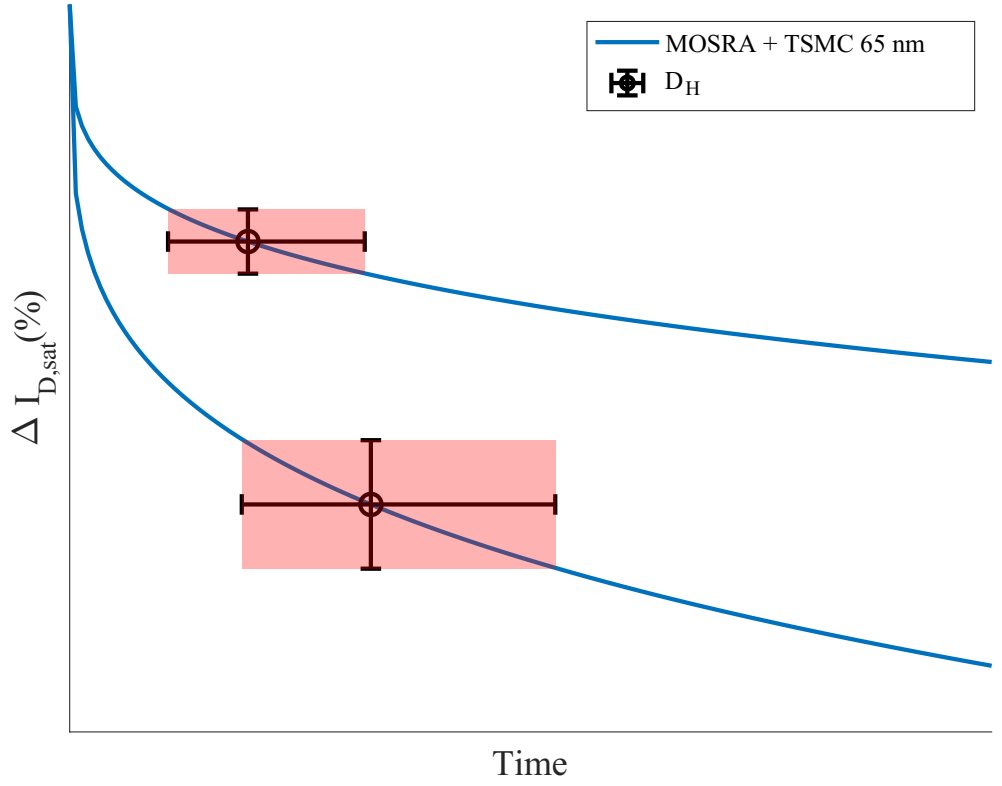


Figure 5.8: Demonstration of testing the bounds on the accuracy of our HCI model (blue curves). Measured data (D_{H}) is mapped onto our simulation curves and the horizontal error bars are checked for overlap. The D_{H} points are assumed to have occurred at the same age. If their time error bars overlap then the lower bound on the accuracy of our model is equal to the accuracy of the M_{H} model.

This chapter has presented the development of an iterative method to fit a compact MOSFET degradation model for use with a specific target technology. Data used to fit a model is aggregated from multiple public domain sources. This is made possible through an assumption that HCI degradation effects generalize across similar technologies. Results are validated against time to failure statistics from the target technology's reliability rules. The method was demonstrated by fitting Synopsys' compact HCI model for use with both core and I/O nMOSFETs from TSMC's 65 nm technology with a high degree of confidence. The results can match the accuracy of a previously published model when simulating HCI degradation in an inverter. The developed method for validating the fit to a deterministic model against a statistical specification is general and can easily be adapted for use with

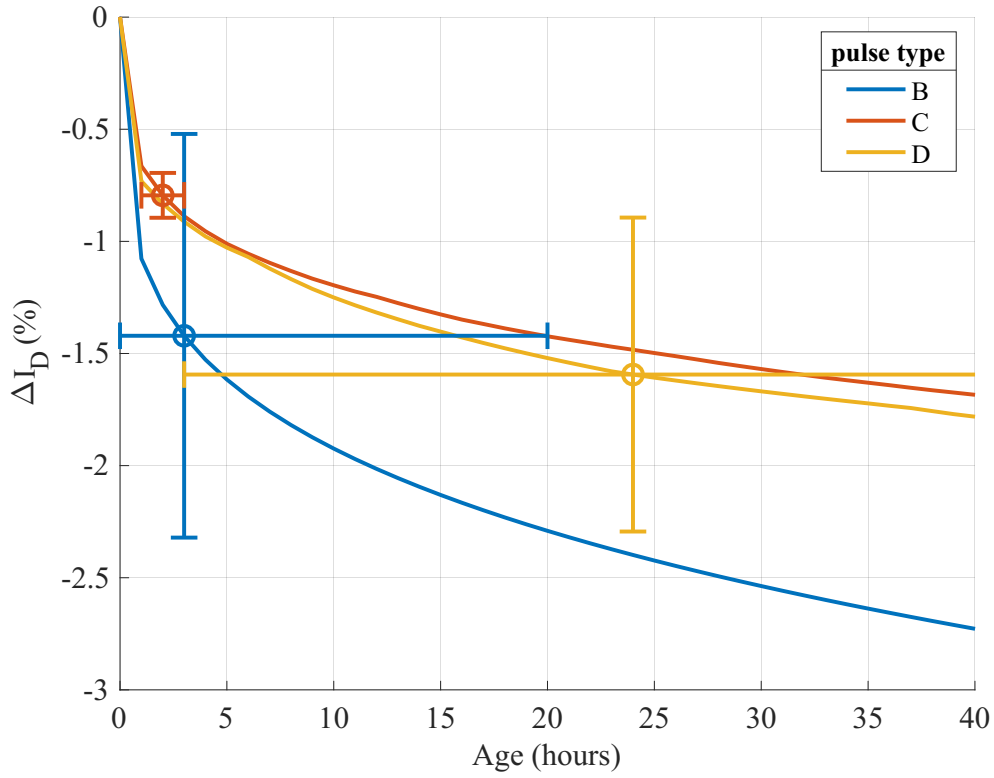


Figure 5.9: Inverter degradation according to MOSRA with $\mathbf{v_{core}}$. V_{DD} is 1.5 V. Also shown are the D_H points, each mapped onto its corresponding curve.

other EDA tools and CMOS technologies. The results from this chapter are applied in the design and validation of the HCI-based anti-recycling sensor presented in the next chapter.

Chapter 6

A Small Tamper-Resistant Anti-Recycling IC Sensor With a Reusable I/O Interface and DC Signalling

CHAPTER 3 argued that embedded sensors are an attractive strategy for mitigating the threat of recycled ICs. Several desirable features for such a sensor, which would present an improvement over the state of the art, were laid out. Among these is the exploitation of HCI to track an IC's age, a strategy which an initial assessment in section 3.2 showed promise.

In this chapter, a sensor with the characteristics described in section 3.1 is presented. The results of chapter 5 are used here to design and assess the performance of this sensor with a high degree of confidence.

As covered in section 2.2, much effort has been spent on developing countermeasures to IC counterfeiting that determine the age of individual ICs with a high precision. This is not an effective strategy for three reasons. First, exhaustively determining the exact powered-

on age of every IC scales poorly and is untenable at real-world scales of tens of thousands to millions of devices. Second, recycled ICs can reasonably be expected to range in age from months to years. This is justified by relying on the average age of mobile phones when discarded by users, which globally is in excess of 20 months [57], as a proxy for the age of recycled components. Third, it makes little sense for a counterfeiter to inject a few recycled components into a batch of ICs. An illegitimate batch is far more likely to be composed entirely, or in majority of counterfeits. The problem at hand is thus to assess with reasonable accuracy whether a purchased batch of components is beyond a chosen powered-on age threshold set well below the expected age of recycled components, e.g., whether a batch is older than 6 months. This allows low-cost, time efficient, random sample-based testing and assessment approaches to be used.

This chapter proposes a novel IC sensor to address the above problem. It differs from the existing IC age sensor literature in the following ways:

1. it has improved tamper-resistance through exploiting hot carrier injection (HCI), the effects of which do not reverse once the responsible stress is removed [32];

2. it produces an DC signal that is easily measurable via standard low-cost testing equipment;

3. it makes use of existing I/O interfaces, thereby requiring no specialized interface circuitry, nor any extra pins;

4. it has a comparatively small footprint, allowing easy integration into an existing IC design;

5. it is designed to allow for low-cost, coarse time resolution age detection suitable to efficiently assess fraudulent reuse within larger volume IC batches.

The rest of this chapter is organized as follows. We present our sensor design in section 6.1 and describe its operation in section 6.2. Section 6.3 demonstrates that through a random

sampling approach, our sensor enables the detection of counterfeit batches having an average age of at least 6 months.

6.1 Sensor Design

Our sensor, dubbed the tamper-resistant HCI-based anti-recycling countermeasure with DC signalling (THARC-DC), is shown in Figure 6.1. To our knowledge, concurrently achieving the features described by our claims 1-4 is unique to our design. Figure 6.1 also shows the integration of the THARC-DC into a typical I/O block consisting of a pin, an electrostatic discharge (ESD) protection composed of grounded gate MOSFETs [58], and a pair of buffers.

The sensor is embedded into an I/O block to:

1. enable access via an existing I/O pin, thus simplifying integration and lowering cost and complexity; and

2. allow for easy access to the higher voltage of the I/O power rail which is used for accelerated aging in the THARC-DC.

THARC-DC has a very simple structure. At its heart is the nMOSFET M_{age} which experiences accelerated HCI degradation. A voltage divider (M_{div1} and M_{div2}) and pMOSFET pass gate (M_{aen}) set the voltages on M_{age}. The sensor is connected to the I/O block through the transmission gate TG_{age}, allowing it to be read at the pin via standard DC testing.

With the Exception of M_{age}, which is a core type device, THARC-DC is entirely comprised of I/O type devices. In this way, we can amplify the HCI stress on M_{age} while ensuring that all other devices experience no appreciable degradation. HCI in M_{age} can be maximized by: 1. minimizing the channel length; and 2. maximizing the gate and drain voltages. By using a minimum-sized core device for M_{age}, its channel is as short as possible.

It is well known that transistors operated at or below their technology's nominal voltages do not experience any appreciable degradation due to HCI or other mechanisms. This is by

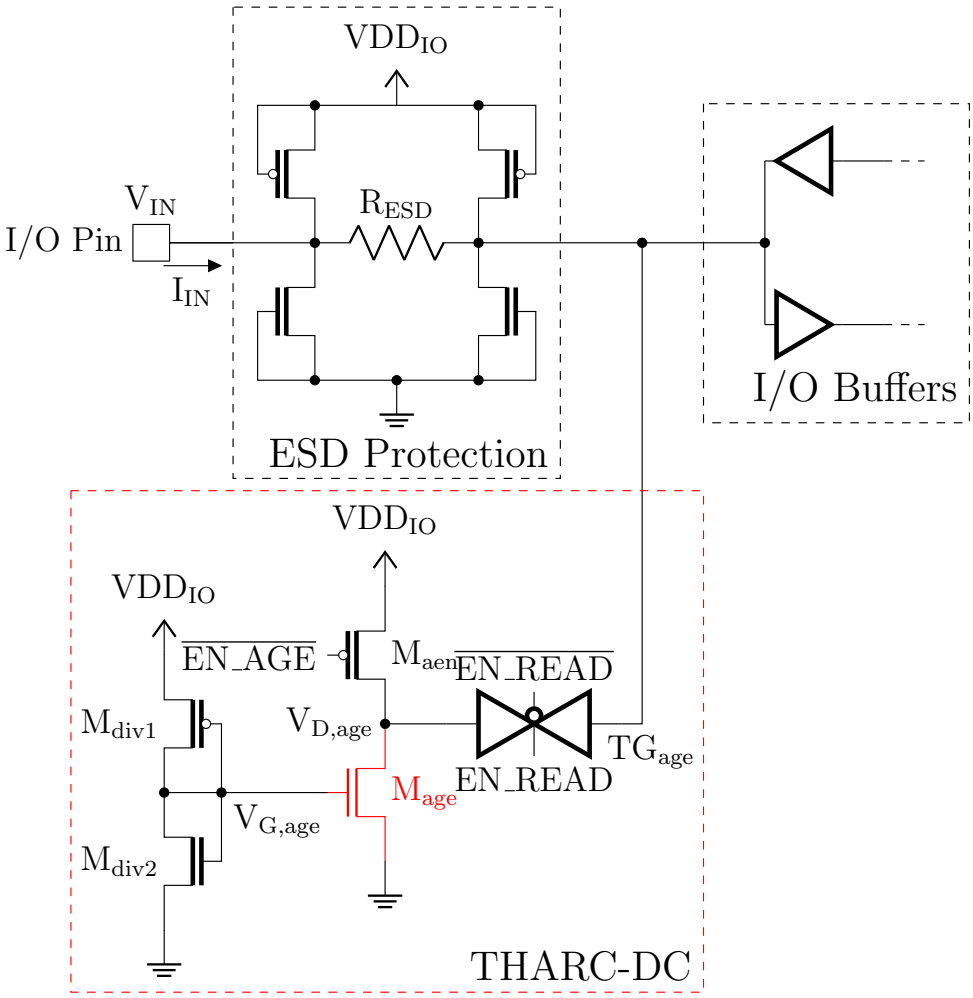


Figure 6.1: THARC-DC embedded into an existing I/O block. It is designed to subject M_{age} to an accelerated HCI degradation. Its state is assessed by the current it draws (I_{IN}) from the pin under a reference DC voltage (V_{IN}). I/O devices are depicted with thicker gates to differentiate them from the core device M_{age}.

design so as to ensure a technology's reliability. Thus to subject M_{age} to a large HCI stress, its drain and gate voltages must exceed VDD_{core}, the nominal core voltage. This is easily done by tapping into the I/O power rail. However, the full I/O voltage, VDD_{IO}, cannot be applied to M_{age}'s gate without significantly increasing the risk of a gate oxide failure. The voltage divider sets $V_{G,age}$ higher than VDD_{core}, but not so high as to endanger the gate oxide. Ideally $V_{D,age}$ should be equal to $V_{G,age}$, so as to allow the maximum possible HCI stress. Though M_{aen} is sized such that $V_{D,age}$ is approximately equal to $V_{G,age}$, an exact match cannot be guaranteed due to process variations.

Design was carried out using TT corner models of a 65 nm technology [64] under nominal conditions of 25 °C and VDD_{IO} of 1.8 V. The lower limit for $V_{G,age}$ is VDD_{core} (1.0 V), while the upper limit is determined by the technology's guidelines for gate oxide integrity. These specify the maximum gate voltage which will ensure that no more than 0.1 % of devices will experience a soft breakdown within 10 years [58]. For core devices, like M_{age}, this voltage is 1.26 V at 65 °C, decreasing to 1.21 V at 105 °C [58]. Setting $V_{G,age}$ to 1.1 V provides a safety margin for M_{age}'s oxide while still allowing for an accelerated HCI degradation. M_{aen} was sized such that $V_{D,age}$ would be as close as possible to $V_{G,age}$, resulting in a value of 1.11 V. The spreads of $V_{G,age}$ and $V_{D,age}$ were determined through a 1000 instance Monte Carlo simulation using global variation models which cover all five process corners (TT, FF, SS, FS, and SF). The results, summarized as the baseline operating condition in Table 6.1, show that at a 3σ deviation $V_{G,age}$ remains within 4 % of the design value and $V_{D,age}$ within 15 %. Additional simulations were performed to cover variations in operating conditions. VDD_{IO} was varied by ±5 % and the temperature increased by 50 °C relative to the baseline condition. The maximum safe voltage for $V_{G,age}$ at 75 °C was determined to be 1.24 V through a cubic spline interpolation to data listed in the technology's guidelines [58]. Results, summarized in Table 6.1, show that under all examined operating conditions, M_{age} is expected to experience accelerated HCI degradation without a decrease in oxide reliability.

Table 6.1: Expected $V_{G,age}$ and $V_{D,age}$ values under the operating conditions: low-voltage, low-temperature (LVLT); low-voltage, high-temperature (LVHT); baseline; high-voltage, low-temperature (HVLT); and high-voltage, high-temperature (HVHT).

operating condition	VDD_{IO} (V)	T (°C)	$V_{G,age}$ (V) $\mu \pm 3\sigma$	$V_{D,age}$ (V) $\mu \pm 3\sigma$
LVLT	1.71	25	1.04 ± 0.04	1.06 ± 0.16
LVHT	1.71	75	1.06 ± 0.04	0.99 ± 0.17
baseline	1.80	25	1.10 ± 0.04	1.11 ± 0.16
HVLT	1.89	25	1.17 ± 0.05	1.16 ± 0.16
HVHT	1.89	75	1.19 ± 0.05	1.09 ± 0.17

None of the I/O devices experience any appreciable HCI degradation since they are not subjected to voltages in excess of VDD_{IO}.

6.2 Sensor Operation

THARC-DC has the mutually exclusive operating modes of: **AGE**, for applying HCI stress to M_{age}, and **READ**, for assessing the accumulated HCI degradation of M_{age}. These modes are controlled respectively by the signals EN_AGE and EN_READ. **AGE** mode is concurrent with normal system operation so that IC powered-on age may be tracked. On the other hand, **READ** mode cannot be active during normal system operation since THARC-DC reuses an existing I/O pin and data collisions must be avoided. This does not present a problem as THARC-DC is intended to be read during standard DC testing arising between the IC's purchase and deployment. Additionally, the measurement benefits from the quieter electrical environment resulting from the lack of activity in the IC. As a result, EN_AGE and EN_READ can be tied to an existing control signal such as chip enable, clock enable, or reset, and as such do not need any dedicated control pins of their own. Depending on the exact control signals used, level shifters may be required since M_{aen} and TG_{age} are I/O type devices. Both EN_AGE and EN_READ can be tied to the same signal since **AGE** and **READ** are mutually exclusive. In the following discussion, the sensor is assumed to

be controlled by a system reset signal, named RESET, such that $\overline{\text{EN\_AGE}}$ = RESET and EN_READ = RESET.

During normal system operation, RESET = 0 and THARC-DC is kept in **AGE** mode. The pass gate M_{aen} is activated ($\overline{\text{EN\_AGE}} = 0$) and the transmission gate TG_{age} is deactivated (EN_READ = 0). This sets $V_{D,age}$ to a voltage higher than VDD_{core}, ensuring that M_{age} is subjected to an enhanced HCI stress. It also isolates the sensor from the rest of the I/O block so that it will not interfere with normal system operation.

When an end-user decides to place THARC-DC in **READ** mode, they set and maintain RESET = 1. This deactivates M_{aen} ($\overline{\text{EN\_AGE}} = 1$) and activates TG_{age} (EN_READ = 1). The drain of M_{age} is disconnected from VDD_{IO} and connected to the I/O pin through the ESD protection. To read the sensor's state, an external DC voltage source V_{IN} is connected to the pin and I_{IN}, the resulting current drawn by M_{age}, is measured. This easy measurement can even be performed using any commonly available bench-top source measure unit (SMU). With passing time, I_{IN} decreases due to the increasing HCI degradation of M_{age} accumulated during **AGE** mode. This very simple usage model is an intentional aspect of the sensor's design.

6.3 Results and Discussion

We simulated THARC-DC along with a typical ESD protection, as depicted in Figure 6.1, using Synopsys HSPICE [62] and a foundry process design kit (PDK) for a 65 nm technology [64]. R_{ESD} was set to 200 Ω [58] and global variation models covering the five process corners were used for all transistors. Separate Monte Carlo simulations, each consisting of 1000 instances, were performed to assess THARC-DC performance under each of the operating conditions listed in Table 6.1. Operating conditions were varied during **AGE** mode only. Since an end-user would have complete control over measurement conditions, the baseline operating condition was used during **READ** mode in all cases. HCI effects were

only simulated in nMOSFETs since PMOSFETs are impacted to a far lesser extent [30]. As discussed in chapter 4, running MOSRA simulations with Monte Carlo is cumbersome when the post-stress simulation conditions differ from the pre-stress conditions. The workflow, depicted in Figure 6.2, was automated with a custom shell script.

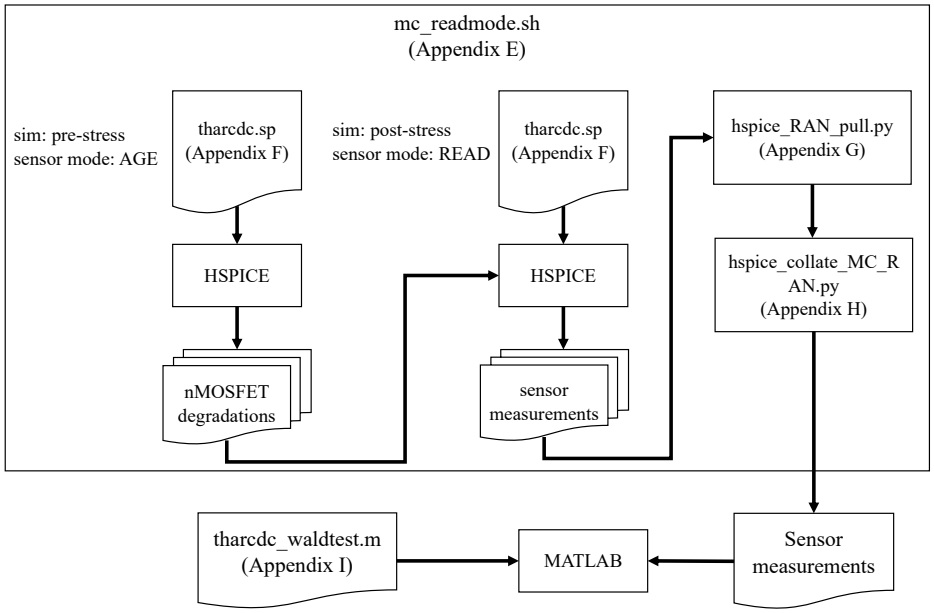


Figure 6.2: Simulation and analysis workflow for THARC-DC.

Predicted typical HCI effects on M_{age} over 10 years under baseline operating conditions are shown in Figure 6.3. While $V_{D,age}$ is pulled towards VDD_{IO} due to the diminishing voltage drop across M_{aen} as $I_{D,age}$ falls, $V_{G,age}$ is held nearly constant by the voltage divider and M_{age} is protected from a premature oxide failure. This same behavior was observed under the various operating conditions listed in Table 6.1. After 10 years the increase in $V_{G,age}$ was less than 0.2 % in all cases.

Figure 6.4 shows the evolution of the average static current in THARC-DC under different operating conditions. The majority of the current is attributable to M_{age} and decreases as this device experiences HCI degradation. Depending on the operating condition, the initial static current ranges from 580 µA to 760 µA, which represents less than 0.2 % of the average quiescent current consumed by a 65 nm node FPGA family [75].

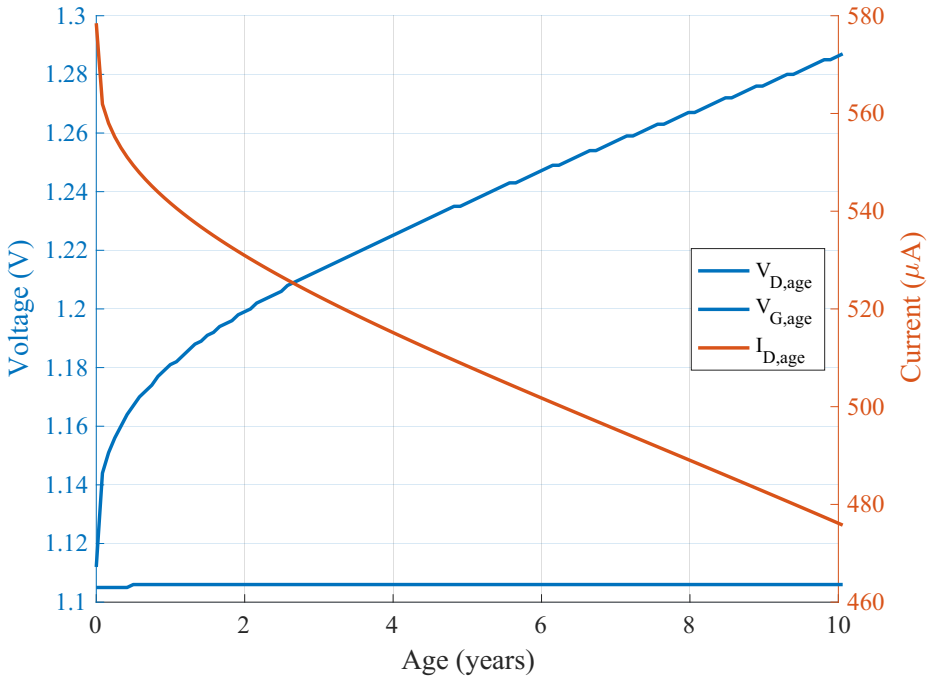


Figure 6.3: Predicted M_{age} degradation due to HCI while the sensor is operating in **AGE** mode under the baseline operating condition (VDD_{IO} of 1.8 V and a temperature of 25 °C). Results are from TT corner models.

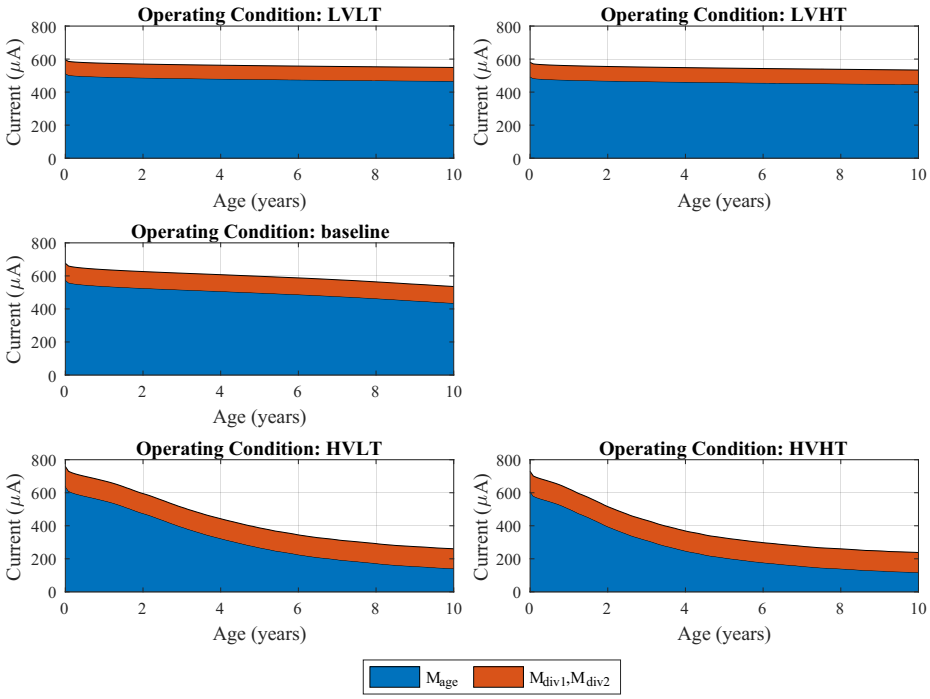
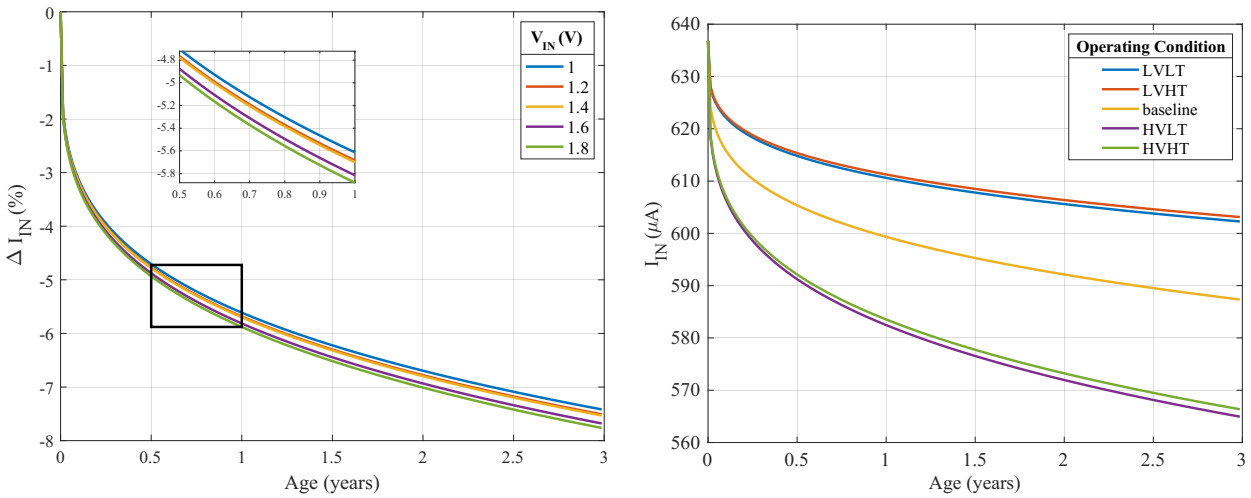


Figure 6.4: Average static current in THARC-DC over time. Operating conditions are those defined in Table 6.1.

Figure 6.5 shows typical **READ** mode current measurements, all of which were performed under the baseline operating condition regardless of the operating condition during **AGE** mode. Figure 6.5(a) compares relative I_{IN} degradation in response to different V_{IN} levels after aging under the baseline operating condition. Increasing V_{IN} leads to an increase in the measured degradation signal. For example, after one year I_{IN} will be measured to have degraded by 5.6 % when V_{IN} is 1.0 V and 5.9 % when V_{IN} is 1.8 V, a relative difference of about 5 %. The highest possible V_{IN} should therefore be used when measuring THARC-DC. Figure 6.5(b) shows I_{IN} measurements using $V_{IN} = 1.8$ V after aging under different operating conditions. THARC-DC is impacted to a far greater extent by variations in VDD_{IO} than by variations in temperature. The impact on the degradation of I_{IN} due to a 5 % variation in VDD_{IO} is an order of magnitude larger than the impact due to a tripling of the temperature.



(a) **(b)**

Figure 6.5: I_{IN} under **READ** mode. All measurements were performed under the baseline operating condition. (a) Relative degradation under different input voltages after aging under the baseline operating condition. (b) Degradation after aging under the different operating conditions defined in Table 6.1. Results are from typical corner models.

The results shown in Figure 6.5 are idealized since they were generated using typical corner models which do not account for manufacturing process variations [64]. For real sensors these variations must be taken into account, consequently the sensor measurements

are described by the stochastic process $I_{\mathrm{IN}}(t)$. At a given age t, $I_{\mathrm{IN}}(t)$ is a random variable drawn from a distribution with a mean $\mu(t)$ and variance $\sigma^2(t)$, both of which exist but are assumed unknown.

Authenticating a batch of suspicious ICs, each containing a single THARC-DC, therefore consists of testing the hypothesis that $I_{\mathrm{IN}}(t)$ is drawn from the same distribution as $I_{\mathrm{IN}}(0)$. To simplify the situation for the end-user, we focus on the mean rather than the entire distribution. The Wald test is well-suited to testing the null hypothesis $H_0 : \mu(t) = \mu(0)$ [53]. The size α Wald test rejects H_0 if $\mu(0) \notin C$ where

$$C = \left(\hat{\mu}_n(t) - \frac{\hat{\sigma}_n(t)}{\sqrt{n}} z_{\alpha/2}, \hat{\mu}_n(t) + \frac{\hat{\sigma}_n(t)}{\sqrt{n}} z_{\alpha/2} \right) \tag{6.1}$$

is the $1 - \alpha$ confidence interval on $\mu(t)$. Computing C requires the sample mean

$$\hat{\mu}_n(t) = \frac{1}{n} \sum_{i=1}^{n} I_{\mathrm{IN},i}(t), \tag{6.2}$$

the sample standard deviation

$$\hat{\sigma}_n(t) = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} \left(I_{\mathrm{IN},i}(t) - \hat{\mu}_n \right)^2}, \tag{6.3}$$

the sample size n, and the $\alpha/2$ upper quantile of the standard Gaussian distribution $z_{\alpha/2}$. The only information not available to an end-user is $\mu(0)$. This must be measured by an IC manufacturer who can then easily publish it in a data sheet.

Minimizing the sample size is important for economical testing; however, too small an n can increase the chance of an incorrect authentication of the batch in question. The situation is illustrated in Figure 6.6. The sample size determines how well $\hat{\mu}_n(t)$ estimates $\mu(t)$, as evidenced by $|C| \propto 1/\sqrt{n}$. As n decreases, the confidence interval on $\mu(t)$ widens and eventually encompasses $\mu(0)$. For every $t > 0$ there is a minimum sample size $n_{min}(t)$ for which $\mu(0) \notin C$. If $n < n_{min}(t)$, the Wald test will fail to reject H_0 and the batch will

be erroneously accepted as authentic. For small t, $n_{min}(t)$ may be quite large, resulting in a practical lower limit on t below which a fraudulent batch is not detectable. As t increases, $\mu(t)$ moves away from $\mu(0)$, resulting in a decreasing $n_{min}(t)$.

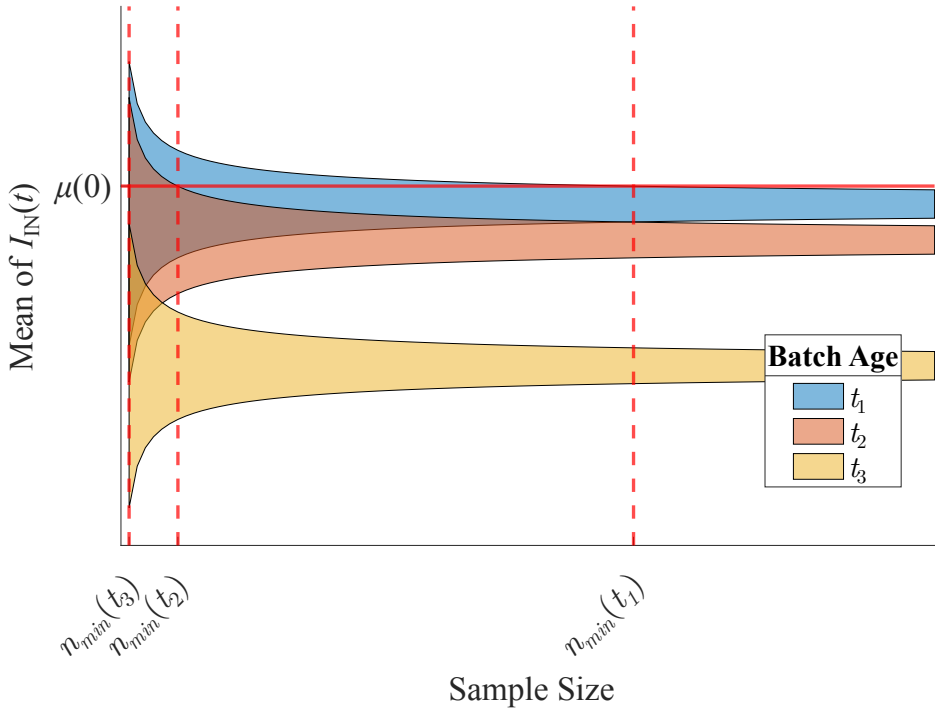


Figure 6.6: Illustration of the Wald test. Shaded regions show confidence intervals on $\mu(t)$ as a function of sample size. Confidence intervals for batches of three different ages $0 < t_1 < t_2 < t_3$ are shown. The Wald test rejects $H_0 : \mu(t) = \mu(0)$ if $\mu(0)$ falls outside of the shaded region. If $n < n_{min}(t)$, the Wald test will incorrectly fail to reject H_0 resulting in an erroneous authentication of the batch.

Figure 6.7 shows the minimum detectable batch age as a function of $n_{min}(t)$ for different operating conditions. For a given $n_{min}(t_i)$, the Wald test is expected to reject H_0 for any batch of age $t \geq t_i$. To produce Figure 6.7, a batch of 1000 THARC-DC was generated and aged over 10 years in 5-day steps through a Monte Carlo simulation using global variation models from the foundry PDK [64]. For each time step a sequence of random samples ranging in size from 50 to 1000 were drawn from the batch and each used to compute $C^{0.99}_{\mu(t),n}$, the 99 % confidence interval on $\mu(t)$ using n samples. Values for $C^{0.99}_{\mu(t),n}$ computed in this way depend on a specific draw of data. In order to find values for $n_{min}(t)$ which generalize, the variance of $C^{0.99}_{\mu(t),n}$ must be estimated. This was done by constructing 99 % bootstrap

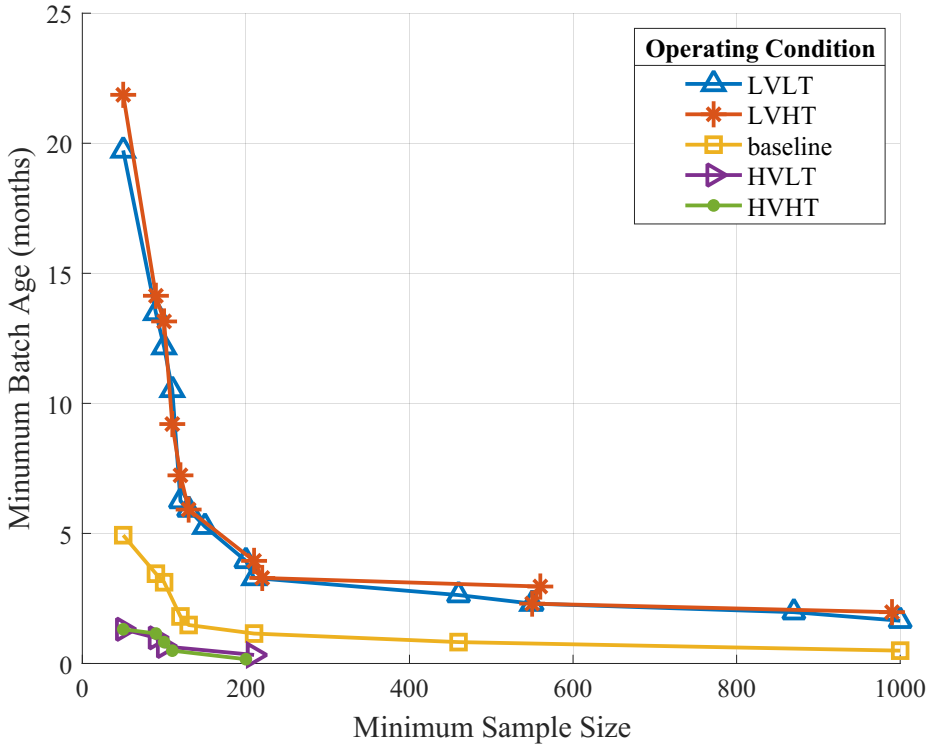


Figure 6.7: Minimum identifiable batch age as a function of sample size and operating condition of the batch. Operating conditions are those defined in Table 6.1. Results are for Wald tests of size 0.01.

pivotal confidence intervals [53] for both $\min\left(C^{0.99}_{\mu(t),n}\right)$ and $\max\left(C^{0.99}_{\mu(t),n}\right)$ through a 1000-fold bootstrap resampling. These new confidence intervals are expected to contain $C^{0.99}_{\mu(t),n}$ 99 % of the time a random sample is drawn from the batch. These were then used to find $n_{min}(t)$. This process was repeated for each operating condition listed in Table 6.1.

Figure 6.7 shows results for homogeneous batches; they are composed of ICs of the same age and which have experienced the same constant operating conditions. In reality, batches of suspect ICs can only be expected to be homogenous if they are in fact genuine. Nonetheless, our results demonstrate that our technique can identify a counterfeit batch composed of ICs of mixed, indeterminate ages with a reasonable sample size. Consider a suspect batch with an average age t_{avg} and consisting of ICs which have operated under a variety operating conditions. The least and most stressful expected operating conditions are LVHT (low-voltage, high-temperature) and HVLT (high-voltage, low-temperature). Therefore the homogenous batch which has operated under LVHT for t_{avg} will have undergone the least possible average degradation and require the largest $n_{min}(t_{avg})$. Consequently, a Wald test performed using this sample size can be expected to reject H_0 for any batch with an average age of at least t_{avg}. Our results show that a batch with an average age of at least 6 months, which we believe to be a reasonable lower bound on the age of fraudulently recycled devices, is detectable at a statistical significance level of $\alpha = 0.01$ by testing a random sample as small as 130. The detectable minimum average batch age decreases to 3 months with a sample size of 500, and to 2 months with a sample size of 1000. Depending on the specific operating conditions experienced by the constituent ICs, younger batches may be detectable with these sample sizes. The detection method may be defeated by a counterfeiter who ensures that the average I_{IN} is kept close to $\mu(0)$. This is unlikely as it would require a batch to contain a large number of genuine ICs, which runs counter to the economic motivation driving the counterfeiting.

It is impossible to prove that H_0 is correct, it can only be shown to be incorrect through sufficient evidence [53]. The goal of authentication is therefore to search for evidence that a

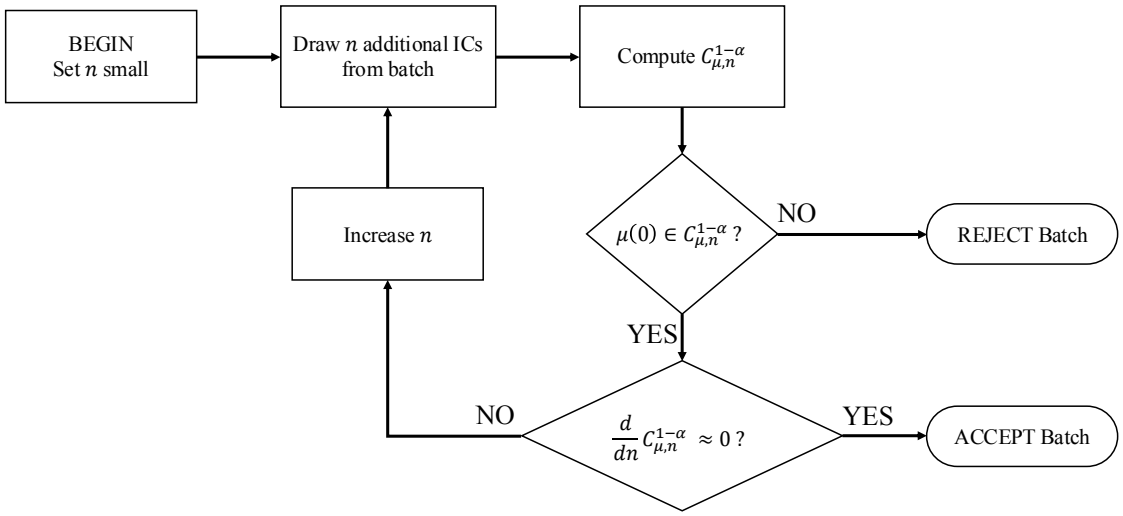


Figure 6.8: THARC-DC authentication process.

batch is not authentic. An efficient strategy, depicted in Figure 6.8, is to iteratively perform the Wald test with an increasing n. Because $C_{\mu,n}^{1-\alpha} \propto 1/\sqrt{n}$, the rate at which the confidence interval narrows decreases with n. Eventually, $C_{\mu,n}^{1-\alpha}$ will either move away from $\mu(0)$ or it will become essentially constant with n. In the former, H_0 is rejected and the batch can be judged to be fraudulent. In the latter, testing can cease and the conclusion that no evidence of fraud is likely to be found may be drawn. Figure 6.7 can aid an end-user in deciding when to accept a batch as being likely authentic. For example, increasing the sample size beyond 500 provides little benefit and it may be decided to accept a batch if H_0 is not yet rejected at this point. The authentication process can be performed without access to the information in Figure 6.7; however, an IC manufacturer could easily support end users by providing it in a data sheet.

This chapter has presented the development of THARC-DC, a novel on-chip anti-recycling sensor. By exploiting HCI, our sensor has improved tamper-resistance as compared to previously proposed BTI-based designs. THARC-DC is queried through a DC measurement which requires only commonly available, low-cost, low-effort equipment and processes. This simple measurement allows for a novel interface that piggy-backs onto an existing I/O interface and requires no additional pins or specialized control signals. Due to its small footprint, our

sensor can be easily integrated into many systems. Previously proposed anti-counterfeiting sensors were designed with the goal of authenticating individual ICs, a strategy with poor scalability. We have proposed instead to use THARC-DC in conjunction with random testing to rapidly authenticate entire batches of suspect ICs. By measuring successively larger random samples the status of a batch can be determined with the desired level of statistical significance. Due to the simple DC measurement involved, this authentication strategy is accessible to organizations ranging in size from international corporations down to small startups.

Chapter 7

Conclusions

THE landscape of the modern semiconductor industry is highly complex. Vertically integrated companies have been largely replaced by niche firms, resulting in a mostly horizontally integrated and globally distributed industry. The ensuing supply chains are convoluted to the point that they cannot be fully trusted, which has created openings for counterfeiters. Counterfeiting by recycling, where components recovered from waste streams are refurbished and sold as new, is a particularly vexing problem that has been identified by the U.S. Department of Defense as the greatest risk to the reliability of its systems. This is due to the true state of recycled components being unknown at the time of their integration, leading to unpredictable system failures. Unlike consumer products, which evolve rapidly and are usually thrown away after a few years, critical infrastructure in areas such as health care, transportation, and defense must be maintained over operating lifetimes stretching into decades. Sourcing replacement components for these systems can be challenging because consumer applications now completely dominate the semiconductor market and it is no longer economically feasible to have dedicated production in the service of critical long-lived infrastructure. As a result, these sectors are dependent on supply chains which are completely intertwined with those for consumer products, making them susceptible to recycled counterfeits. This problem will only continue to worsen as the global production of

electronics, and the resulting electronic waste, continues to increase. A likely new and consequential destination for recycled components is the burgeoning IoT devices being integrated into everything from traffic control to medical aids.

The problem of recycled ICs is challenging due to its asymmetrical nature. Producing this type of counterfeit is easy and inexpensive whereas detecting it is difficult. The only accepted detection methods involve physical inspection, which is slow and dependent on expert interpretation. Many countermeasures have previously been proposed, a number of which were reviewed in chapter 2. They fall into two broad categories: preemptive methods, which aim to ease counterfeit detection by modifying an IC design before fabrication, typically by including a specialized sensor; and passive methods, which seek to identify counterfeits purely through non-destructive measurements performed on ICs. They suffer from a number of issues, which were summarized in chapter 3:

- Embedded sensors must communicate with the outside world and proposed designs do not have an elegant solution. Some require a dedicated pin, others propose to use a JTAG interface, others yet do not fully address the issue. If the JTAG interface is to be used, a controller for sensor is required. Some designs include a nonvolatile memory to keep track of an age counter. These features add complexity to the sensor and increase the cost and difficulty of integration into an IC design.

- The targeted degradation mechanism is difficult to harness for properly assessing the age of a sensor. For example, bias temperature instability begins to reverse once an IC is powered down, and electromigration is stochastic.

- Measurements may be complicated and lengthy. This is especially true for proposed passive methods such as measuring specific path delays.

- A golden reference device is required for a number of passive methods. Alternatively, complex information, such as path delays and corresponding test patterns or degrada-

tion curves, drawn from a golden reference are required. How this information is to be provided to an end-user is not addressed.

- Many passive methods relying on parametric measurements use a classifier to identify recycled ICs. However, the methods are sensitive to process variations and the classifier would need to be retrained any time there was a change in the production line. Who would retrain the classifier, and on which data is not addressed.

- Some methods, such as light emission due to off-state leakage current, can facilitate reverse engineering and pose a threat to the IC designer's intellectual property.

These shortcomings leave room for the more effective approach to combatting IC recycling presented in this dissertation.

7.1 Summary of Work

The major contributions presented in this dissertation are:

- A general method for fitting a MOSFET degradation model to a target technology by leveraging public domain data. The solution produces results which are statistically validated against the target technology's specifications. This contribution was published in the IEEE Access Journal [13].

- A novel compact IC sensor for the identification of IC batches containing recycled counterfeits. This contribution was published in the IEEE Open Journal of Circuits and Systems [14] and an initial feasibility study of the idea was presented at the IEEE 61st International Midwest Symposium on Circuits and Systems [15].

As was argued in chapter 3, an embedded sensor is an attractive strategy because it can be purposefully designed to generate a clear and easily measured aging signal. Though several sensor have been proposed, improvements over the state of the art are possible.

Proposed designs have sought to authenticate individual ICs through accurate and precise determination of their ages. This is wasted effort for two reasons. First, random sampling methods would be used in any realistic scenario since exhaustive testing would be enormously expensive at large scales. Second, recycled ICs are likely to range in age on the scale of months to years so only a rough idea of their ages should suffice for authentication. In this work, a different strategy was followed, where a sensor was designed for the authentication of an entire batch of ICs through random sampling. This sensor, THARC-DC, was presented in chapter 6 and has the following novel features:

1. It exploits hot carrier injection (HCI) for improved tamper resistance over previously suggested sensors that exploit BTI.

2. It produces a DC signal that is easily measurable via standard low-cost testing equipment.

3. It requires no extra pins, nor any specialized interface circuitry because it makes use of existing I/O interfaces.

4. It can be easily integrated into existing IC designs as a consequence of its small footprint.

5. It allows for low-cost, coarse time resolution age detection suitable to efficiently assess fraudulent reuse within larger volume IC batches.

THARC-DC can be used to efficiently search for evidence that a batch is fraudulent by iteratively testing successively larger samples drawn from the batch in question. As the sample size is increased, the confidence interval on the mean THARC-DC measurement narrows and younger fraudulent batches become detectable. To perform an authentication on a batch of suspect ICs which contain THARC-DC, an end-user would begin by drawing a small random sample, perhaps a few dozen, from the batch. They would then measure THARC-DC output from each IC in the sample under uniform conditions. Next, they would

perform a Wald test, as described in chapter 6, to compare the average sample measurement to the reference value published in the IC's data sheet. If the test results in a rejection of the null hypothesis, there is evidence that the batch is not authentic. Otherwise, they would randomly draw more ICs from the batch and repeat the test using the expanded sample. They would continue in this fashion with increasingly large samples until either the null hypothesis is rejected, or the sample size grows impractically large. In the latter case, the user could conclude that evidence of the batch's fraudulence is unlikely to be found and may then decide to accept the batch. Due to the simple DC measurement involved, this authentication strategy is accessible to organizations ranging in size from international corporations down to small startups. Simulation results show that when implemented in a TSMC 65 nm technology, THARC-DC permits the detection of a batch of recycled ICs with an average age of at least 6 months at a statistical significance level of $\alpha = 0.01$ by testing a random sample as small as 130.

THARC-DC was designed and simulated using Synopsys HSPICE. Critical to this work is the MOSRA facility of HSPICE, which simulates the effects of HCI on circuit performance. In order for MOSRA to produce reliable results, its model parameters must be tuned for the specific targeted technology. MOSFET degradation data is required for this. Without access to confidential fab data and without the resources to fabricate and measure a suite of test devices, an alternate approach was needed. Chapter 5 presented an iterative method developed to fit a compact MOSFET degradation model for use with a specific target technology. Data used to fit a model is aggregated from multiple public domain sources. This is made possible through an assumption that HCI degradation effects generalize across similar technologies. Results were validated against time to failure statistics from the target technology's reliability rules. The method was demonstrated by fitting Synopsys' compact HCI model for use with both core and I/O nMOSFETs from TSMC's 65 nm technology with a high degree of confidence. The results were also shown to match the accuracy of a previously published model when simulating HCI degradation in an inverter. This method

for validating the fit to a deterministic model against a statistical specification is general and can easily be adapted for use with other EDA tools and CMOS technologies.

There are several limitations associated with the method. Node size alone is an insufficient measure of similarity between technologies when selecting data sources, some care is therefore required. Ultimately the accuracy of any predictions born from the results is limited in two ways. First, the error in these model parameter values cannot be measured without access to a golden reference data set. This would have to be provided by a fab, or measured from a set of test devices. Second, the MOSRA model lacks the saturation mechanisms necessary to produce accurate long time or extreme operating condition degradation estimates. Regardless, the derived results are adequate for generating circuit performance predictions, particularly at initial design phases. This is of particular interest to anyone seeking to minimize the cost of exploratory design work, such as academics.

7.2 Future Work

This dissertation has presented a viable countermeasure against recycled ICs. As with any research project, further refinements and developments are possible. Described below are several such possibilities.

The MOSRA HCI model parameters fitted in chapter 5 were both derived and validated using incomplete information. Proper validation of the results requires the fabrication and measurement of a set of real test devices in TSMC's 65 nm technology. This would first involve deriving a new set of model parameters from these test measurements. Any ambiguity would be eliminated from the fitting process since the test devices would be designed to properly cover all dimensions of the model. Next, the two models would be used to generate predictions of nMOSFET degradation for comparison to real test device measurements. In this way, the inherent error of the MOSRA model could be characterized as well as the error in the fitting method described in this work.

Adapting the work presented this dissertation to more advanced nodes presents a new avenue of inquiry. The MOSRA HCI model used in this work may not be applicable to more modern technology nodes, especially those using non-planar geometries. It may be as simple as fitting the MOSRA model to a different data set. On the other hand it may involve adopting an entirely different HCI model, which may have implications for the fitting process as well as the design of THARC-DC.

It was found in section 5.3 that the MOSRA HCI model cannot be fit to an arbitrary aggregation of data whose only common characteristic is the technological node size of the devices from which it was measured. The implication is that technological factors other than node size play an important role in HCI degradation. It may be possible to use HCI degradation in the creation of fingerprints to identify the fab from which an IC originated. This could lead to the creation of a countermeasure against a second class of counterfeits, illegitimately produced ICs, in addition to recycled ICs.

Several avenues of future work based the THARC-DC sensor are possible. Performance estimates in chapter 6 were generated using a worst-case scenario, where the suspect ICs had been operated at a reduced VDD_{IO} and elevated temperature. Performance estimates of more realistic scenarios are possible. This would require generating batches of mixed-age ICs operated under non-uniform conditions. For more realism, the ages should not be drawn from a uniform distributed but rather from a lognormal distribution.

Sensor aging was not greatly affected by variations in operating temperature. This was not the case for changes in supply voltage. As shown in Figure 6.1, the gate voltage of M_{age} is generated by a simple voltage divider. The design may be improved by replacing this divider with a more robust voltage source.

The transistor targeted for accelerated HCI degradation, M_{age}, is a regular nMOSFET. Like all such modern devices, it is constructed with a lightly doped drain (LDD), where the drain and source doping concentrations adjacent to the channel are reduced [27]. This feature reduced the saturation electric field between the drain and source and serves to limit

HCI damage. If M_{age} was fabricated without a LDD, it would be subjected to greater HCI degradation. This would result in a stronger aging signal and improved detectability of a recycled IC batch. Modifying THARC-DC in this way would require a new SPICE model for M_{age} as well as a new set of MOSRA model parameters for it. This may be possible through simulation using a technology CAD tool and a sophisticated physics-based HCI model.

Another possible method for enhancing the measured HCI effect is to exchange the drain and source of M_{age} during measurement. HCI damages the gate oxide, which reduces its effectiveness as a dielectric. This increases in the threshold voltage, leading to a decrease in drain current. The oxide is not damaged uniformly along the channel's length, it is concentrated near the drain where the electric field is strongest. When inducing inversion in the channel, charge is first accumulated near the source. If the oxide damage was concentrated near the source instead of the drain, the threshold voltage would be further increased. Thus, the measured HCI degradation could be increased if the source and drain of M_{age} were functionally exchanged during measurements. This would require a more complex sensor design, but may lead to a much improved performance.

A minimum size device was used for M_{age} since a short channel increases HCI degradation, and a narrow channel decreases static current consumption. Increasing the width of this device could provide two advantages. Wider devices have a smaller variance, this could improve detection of recycled devices by reducing the required sample size for the Wald test and/or lower the minimum detectable average batch age. The resulting larger drain current in M_{age} would also be easier to measure. Though a wider device would increase the static current consumption of THARC-DC, this would still likely represent less than 1 % of the static current in a large system such as an FPGA.

THARC-DC measurements could be further simplified for end users if the sensor contained a self-reference combined with a differential sensing scheme. Instead of relying on a single stressed nMOSFET there would be a pair of devices, each experiencing a different HCI stress. The devices would be designed such that the difference in their drain currents

would reach a zero crossing at an intended age. The output signal would therefore be an unambiguously interpreted binary value generated by a saturated differential amplifier. Beyond the simplified measurement, a clear advantage would be the elimination of the need for the manufacturer to publish a reference value in the data sheet. This would however come at the cost of an increase in complexity and footprint for the sensor.

Bibliography

[1] OECD and European Union Intellectual Property Office, *Global Trade in Fakes: A Worrying Threat* (Illicit Trade). OECD, Jun. 22, 2021. DOI: 10.1787/74c81154-en.

[2] J. McBride, "How canada is failing to keep dangerous counterfeit products off store shelves," *Report on Business Magazine*, Nov. 2017. Accessed: Mar. 27, 2018. [Online]. Available: https://www.theglobeandmail.com/report-on-business/rob-magazine/how-canada-is-failing-to-keep-dangerous-counterfeit-products-off-storeshelves/article36714272/.

[3] A. Mutschler. "The Threat Of Supply Chain Insecurity," Semiconductor Engineering, Accessed: Mar. 21, 2025. [Online]. Available: https://semiengineering.com/the-threat-of-supply-chain-insecurity/.

[4] European Union Intellectual Property Office, "Intellectual property crime threat assessment 2022," Luxembourg, 2022. [Online]. Available: https://data.europa.eu/doi/10.2814/830719.

[5] U.S. Senate Committee on Armed Services, "Inquiry into counterfeit electronic parts in the department of defense supply chain," Washington, DC, USA, 112-167, May 21, 2012.

[6] U. Guin, K. Huang, D. DiMase, J. M. Carulli, M. Tehranipoor, and Y. Makris, "Counterfeit integrated circuits: A rising threat in the global semiconductor supply chain,"

Proceedings of the IEEE, vol. 102, no. 8, pp. 1207–1228, Aug. 2014. DOI: 10.1109/JPROC.2014.2332291.

[7] I. Hill, P. Chanawala, R. Singh, S. A. Sheikholeslam, and A. Ivanov, "CMOS reliability from past to future: A survey of requirements, trends, and prediction methods," *IEEE Trans. Device Mater. Reliab.*, vol. 22, no. 1, pp. 1–18, Mar. 2022.

[8] K. Wiens and G. Gordon-Byrne, "The fight to fix it," *IEEE Spectrum*, vol. 54, no. 11, pp. 24–29, Nov. 2017. DOI: 10.1109/MSPEC.2017.8093797. Accessed: Mar. 28, 2018.

[9] "State of IoT 2024: Number of connected IoT devices growing 13% to 18.8 billion globally," IoT Analytics, Accessed: Mar. 22, 2025. [Online]. Available: https://iot-analytics.com/number-connected-iot-devices/.

[10] "IoT connections forecast – Ericsson Mobility Report," ericsson.com, Accessed: Mar. 22, 2025. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/mobility-report/dataforecasts/iot-connections-outlook.

[11] S. Ray, A. Basak, and S. Bhunia, "Patching the internet of things," *IEEE Spectrum*, vol. 54, no. 11, pp. 30–35, Nov. 2017. DOI: 10.1109/MSPEC.2017.8093798.

[12] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for Smart Cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, Feb. 2014. DOI: 10.1109/JIOT.2014.2306328.

[13] A. Dimopoulos, M. Sima, and S. W. Neville, "Leveraging Public Information to Fit a Compact Hot Carrier Injection Model to a Target Technology," *IEEE Access*, vol. 11, pp. 21 417–21 426, 2023. DOI: 10.1109/ACCESS.2023.3251340.

[14] A. Dimopoulos, M. Sima, and S. W. Neville, "A Small Tamper-Resistant Anti-Recycling IC Sensor With a Reused I/O Interface and DC Signalling," *IEEE Open Journal of Circuits and Systems*, vol. 5, pp. 341–348, 2024. DOI: 10.1109/OJCAS.2024.3487072.

[15] A. Dimopoulos, M. Sima, and S. W. Neville, "Novel MOSFET Operation for Detection of Recycled Integrated Circuits," in *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, Windsor, ON, Canada: IEEE, Aug. 2018, pp. 1038–1041. DOI: 10.1109/MWSCAS.2018.8623833.

[16] P. D. T. O'Connor, D. Newton, and R. Bromley, *Practical reliability engineering*. New Delhi: Wiley-India, 2010.

[17] A. W. Strong, Ed., *Reliability wearout mechanisms in advanced CMOS technologies* (IEEE Press series on microelectronic systems), English. Piscataway, NJ : Hoboken, NJ: IEEE Press ; Wiley, 2009, ISBN: 978-0-470-45526-5.

[18] S. More, "Aging degradation and countermeasures in deep-submicrometer analog and mixed signal integrated circuits," Doktor-Ingenieurs, Technische Universität München, Munich, Germany, Apr. 30, 2012, 153 pp. Accessed: Nov. 10, 2017. [Online]. Available: https://mediatum.ub.tum.de/doc/1091978/file.pdf.

[19] S. S. Sapatnekar, "What happens when circuits grow old: Aging issues in CMOS design," in *Proc. 2013 Int. Symp. on VLSI Technology, Systems and Application*, Hsinchu, Taiwan, Apr. 2013, pp. 1–2. DOI: 10.1109/VLSI-TSA.2013.6545621.

[20] P. Pouyan, E. Amat, and A. Rubio, "Process variability-aware proactive reconfiguration technique for mitigating aging effects in nano scale SRAM lifetime," IEEE, Apr. 2012, pp. 240–245. DOI: 10.1109/VTS.2012.6231060.

[21] B. Tudor, J. Wang, W. Liu, and H. Elhak, "MOS device aging analysis with HSPICE and CustomSim," Synopsys, Inc., Mountain View, CA, USA, White Paper, Aug. 2011, p. 5. [Online]. Available: https://www.synopsys.com/content/dam/synopsys/verification/white-papers/mosra-wp.pdf.

[22] H. Mostafa, M. Anis, and M. Elmasry, "NBTI and process variations compensation circuits using adaptive body bias," *IEEE Transactions on Semiconductor Manufacturing*, vol. 25, no. 3, pp. 460–467, Aug. 2012. DOI: 10.1109/TSM.2012.2192143.

[23] M. Igarashi, K. Takeuchi, T. Okagaki, K. Shibutani, H. Matsushita, and K. Nii, "An on-die digital aging monitor against HCI and xBTI in 16 nm Fin-FET bulk CMOS technology," in *Proc. 41st European Solid-State Circuits Conf.*, Graz, Austria, Sep. 2015, pp. 112–115. DOI: 10.1109/ESSCIRC.2015.7313841.

[24] K. Tscherkaschin, T. Hillebrand, M. Taddiken, S. Paul, and D. Peters-Drolshagen, "Temperature- and aging-resistant inverter for robust and reliable time to digital circuit designs in a 65nm bulk CMOS process," in *Proceedings of the 2016 IEEE 22nd International Symposium on On-Line Testing and Robust System Design*, Sant Feliu de Guixols, Spain: IEEE, Jul. 2016, pp. 121–125. DOI: 10.1109/IOLTS.2016.7604683.

[25] D. Rossi, V. Tenentes, S. Yang, S. Khursheed, and B. M. Al-Hashimi, "Aging benefits in nanometer CMOS designs," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 64, no. 3, pp. 324–328, Mar. 2017. DOI: 10.1109/TCSII.2016.2561206.

[26] N. Khoshavi, R. A. Ashraf, and R. F. DeMara, "Applicability of power-gating strategies for aging mitigation of CMOS logic paths," in *Proceedings of the 2014 IEEE 57th International Midwest Symposium on Circuits and Systems*, College Station, TX: IEEE, Aug. 2014, pp. 929–932. DOI: 10.1109/MWSCAS.2014.6908568.

[27] R. J. Baker, *CMOS: circuit design, layout, and simulation* (IEEE Press series on microelectronic systems), 3rd ed. Piscataway, NJ : Hoboken, NJ: IEEE Press ; Wiley, 2010, ISBN: 978-0-470-88132-3.

[28] T. Grasser et al., "Simultaneous extraction of recoverable and permanent components contributing to bias-temperature instability," in *Proc. IEEE Electron Devices Meeting*, Washington, DC, USA: IEEE, Dec. 2007, pp. 801–804. DOI: 10.1109/IEDM.2007.4419069.

[29] T. Grasser et al., "The 'permanent' component of NBTI: Composition and annealing," in *Proc. 2011 IEEE International Reliability Physics Symp.*, Monterey, CA, USA, Apr. 2011, 6A.2.1–6A.2.9. DOI: 10.1109/IRPS.2011.5784543.

[30] T. Grasser, Ed., *Hot Carrier Degradation in Semiconductor Devices*, 1st ed, Cham, Switzerland: Springer International Publishing, 2015.

[31] J. Keane and C. H. Kim, "On-chip silicon odometers and their potential use in medical electronics," in *Proc. 2012 IEEE Int. Reliability Physics Symp.*, Anaheim, CA, Apr. 2012, pp. 4C.1.1–4C.1.8. DOI: 10.1109/IRPS.2012.6241835.

[32] M. Vandemaele, K.-H. Chuang, E. Bury, S. Tyaginov, G. Groeseneken, and B. Kaczer, "The influence of gate bias on the anneal of hot-carrier degradation," in *2020 IEEE International Reliability Physics Symposium (IRPS)*, Apr. 2020, pp. 1–7. DOI: 10.1109/IRPS45951.2020.9128218.

[33] G.-B. Lee, J.-W. Jung, C.-K. Kim, T. Bang, M.-S. Yoo, and Y.-K. Choi, "Improved Self-Curing Effect in a MOSFET With Gate Biasing," *IEEE Electron Device Letters*, vol. 42, no. 12, pp. 1731–1734, Dec. 2021.

[34] D. Son, G.-J. Kim, J. Kim, N. Lee, K. Kim, and S. Pae, "Effect of High Temperature on Recovery of Hot Carrier Degradation of scaled nMOSFETs in DRAM," in *2021 IEEE International Reliability Physics Symposium (IRPS)*, Monterey, CA, USA: IEEE, Mar. 2021, pp. 1–4.

[35] P. Singh, E. Karl, D. Blaauw, and D. Sylvester, "Compact Degradation Sensors for Monitoring NBTI and Oxide Degradation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 9, pp. 1645–1655, Sep. 2012. DOI: 10.1109/TVLSI.2011.2161784.

[36] L. de Paris, G. Posser, and R. Reis, "Electromigration aware circuits by using special signal non-default routing rules," in *Proceedings of the 2016 IEEE International Symposium on Circuits and Systems*, Montreal, QC, Canada: IEEE, May 2016, pp. 2795–2798. DOI: 10.1109/ISCAS.2016.7539173.

[37] M. Cho, C. Tokunaga, M. M. Khellah, J. W. Tschanz, and V. De, "Aging-aware adaptive voltage scaling in 22nm high-k/metal-gate tri-gate CMOS," in *Proceedings of the*

2015 IEEE Custom Integrated Circuits Conference, San Jose, Ca.: IEEE, Sep. 2015, pp. 1–4. DOI: 10.1109/CICC.2015.7338419.

[38] E. Saneyoshi, K. Nose, and M. Mizuno, "A precise-tracking NBTI-degradation monitor independent of NBTI recovery effect," in *2010 IEEE International Solid-State Circuits Conference Digest of Technical Papers*, San Francisco, CA: IEEE, Feb. 2010, pp. 192–193. DOI: 10.1109/ISSCC.2010.5433994.

[39] X. Zhang, N. Tuzzio, and M. Tehranipoor, "Identification of recovered ICs using fingerprints from a light-weight on-chip sensor," in *Proceedings of the 49th Annual Design Automation Conference*, San Francisco, CA: ACM Press, Jun. 2012, pp. 703–708. DOI: 10.1145/2228360.2228486.

[40] U. Guin, X. Zhang, D. Forte, and M. Tehranipoor, "Low-cost on-chip structures for combating die and IC recycling," in *Proceedings of the 51st Annual Design Automation Conference*, San Francisco, CA: ACM Press, Jun. 2014, pp. 1–6. DOI: 10.1145/2593069.2593157.

[41] U. Guin, D. Forte, and M. Tehranipoor, "Design of accurate low-cost on-chip structures for protecting integrated circuits against recycling," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 4, pp. 1233–1246, Apr. 2016. DOI: 10.1109/TVLSI.2015.2466551.

[42] P. F. Newman, "Leakage oscillator based aging monitor," U.S. Patent 7592876B2, Sep. 22, 2009.

[43] M. M. Ahmed et al., "Radiated electromagnetic emission for integrated circuit authentication," *IEEE Microwave and Wireless Components Letters*, vol. 27, no. 11, pp. 1028–1030, Nov. 2017. DOI: 10.1109/LMWC.2017.2750078.

[44] K. He, X. Huang, and S. X.-D. Tan, "EM-based on-chip aging sensor for detection and prevention of counterfeit and recycled ICs," in *Proceedings of the 2015 IEEE/ACM*

International Conference on Computer-Aided Design, Austin, TX: IEEE, Nov. 2015, pp. 146–151. DOI: 10.1109/ICCAD.2015.7372562.

[45] K. He, X. Huang, and S. X.-D. Tan, "EM-Based On-Chip Aging Sensor for Detection of Recycled ICs," *IEEE Design & Test*, vol. 33, no. 5, pp. 56–64, Oct. 2016. DOI: 10.1109/MDAT.2016.2582830.

[46] X. Zhang and M. Tehranipoor, "Design of on-chip lightweight sensors for effective detection of recycled ICs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 5, pp. 1016–1029, May 2014. DOI: 10.1109/TVLSI.2013.2264063.

[47] B. Stamme, "Anti-fuse memory provides robust, secure NVM option," *EE Times*, p. 3, Jul. 5, 2012. Accessed: Dec. 27, 2017. [Online]. Available: https://www.eetimes.com/document.asp?doc_id=1279746.

[48] M. Liu and C. H. Kim, "A powerless and non-volatile counterfeit IC detection sensor in a standard logic process based on an exposed floating-gate array," in *Proceedings of the 2017 Symposium on VLSI Technology*, Kyoto, Japan: IEEE, Jun. 2017, T102–T103. DOI: 10.23919/VLSIT.2017.7998211.

[49] P. Song, F. Stellari, and A. Weger, "Counterfeit IC detection using light emission," in *Proceedings of the 2014 IEEE International Test Conference*, Seattle, WA: IEEE, Oct. 2014, pp. 1–8. DOI: 10.1109/TEST.2014.7035356.

[50] X. Zhang, K. Xiao, and M. Tehranipoor, "Path-delay fingerprinting for identification of recovered ICs," in *Proceedings of the 2012 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, Austin, TX: IEEE, Oct. 2012, pp. 13–18. DOI: 10.1109/DFT.2012.6378192.

[51] M. M. Alam, M. Tehranipoor, and D. Forte, "Recycled FPGA detection using exhaustive LUT path delay characterization," in *Proceedings of the 2016 IEEE International Test Conference*, Fort Worth, TX: IEEE, pp. 1–10. DOI: 10.1109/TEST.2016.7805854. Accessed: Dec. 20, 2017.

[52] K. Huang, Y. Liu, N. Korolija, J. M. Carulli, and Y. Makris, "Recycled IC Detection Based on Statistical Methods," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 6, pp. 947–960, Jun. 2015. DOI: 10.1109/TCAD.2015.2409267.

[53] L. Wasserman, *All of statistics: a concise course in statistical inference* (Springer texts in statistics). New York, NY: Springer, 2004, ISBN: 978-0-387-40272-7.

[54] K. Huang, J. M. Carulli, and Y. Makris, "Parametric counterfeit IC detection via support vector machines," in *Proceedings of the 2012 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, Austin, TX: IEEE, Oct. 2012, pp. 7–12. DOI: 10.1109/DFT.2012.6378191.

[55] Y. Zheng, S. Yang, and S. Bhunia, "SeMIA: Self-similarity-based IC integrity analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 1, pp. 37–48, Jan. 2016. DOI: 10.1109/TCAD.2015.2449231.

[56] A. B. Owen. "Monte carlo theory, methods, and examples," Accessed: Sep. 28, 2023. [Online]. Available: https://statweb.stanford.edu/~owen/mc/.

[57] A. Ng. "Smartphone users are waiting longer before upgrading — here's why," CNBC, Accessed: Oct. 14, 2023. [Online]. Available: https://www.cnbc.com/2019/05/17/smartphone-users-are-waiting-longer-before-upgrading-heres-why.html.

[58] *TSMC 65 nm/ 55 nm CMOS LOGIC/MS_RF DESIGN RULE (CLN65 G/GP/LP/LPG/ULP, CLN55 GP/LP, CMN65 GP/LP, CMN55LP)*, Taiwan Semiconductor Manufacturing Co., LTD, Hsinchu, Taiwan, 2012.

[59] M. Pelgrom, A. Duinmaijer, and A. Welbers, "Matching properties of MOS transistors," *IEEE Journal of Solid-State Circuits*, vol. 24, no. 5, pp. 1433–1439, Oct. 1989. DOI: 10.1109/JSSC.1989.572629.

[60] E. Maricau and G. Gielen, *Analog IC reliability in nanometer CMOS* (Analog circuits and signal processing). New York: Springer, 2013, ISBN: 978-1-4614-6162-3.

[61] R. O. Duda, D. G. Stork, and P. E. Hart, *Pattern Classification*, 2nd ed. New York: Wiley, 2001, ISBN: 978-0-471-05669-0.

[62] *HSPICE user guide: Basic simulation and analysis*, Synopsys, Inc., Mountain View, CA, Jun. 2017.

[63] *MATLAB*, version 9.11.0.1873467, Feb. 3, 2022.

[64] *TSMC 65 nm CMOS RF mixed signal general purpose plus 1P9M+ salicide Cu_lowK 1.0&2.5V SPICE model 65RS*, Taiwan Semiconductor Manufacturing Co., LTD, Hsinchu, Taiwan, 2011.

[65] S. K. Moore, D. Johnson, M. Harris, E. Waltz, P. Patel, and M. Hampson, "The latest developments in technology, engineering, and science: News," *IEEE Spectr.*, vol. 58, no. 8, pp. 5–12, Aug. 2021. DOI: 10.1109/MSPEC.2021.9502948.

[66] *65GPLUS SPICE model usage guide*, Taiwan Semiconductor Manufacturing Co., LTD, Hsinchu, Taiwan, 2011.

[67] V. Huard, C. R. Parthasarathy, A. Bravaix, T. Hugel, C. Guerin, and E. Vincent, "Design-in-reliability approach for NBTI and hot-carrier degradations in advanced nodes," *IEEE Trans. Device Mater. Reliab.*, vol. 7, no. 4, pp. 558–570, Dec. 2007.

[68] G. H. Golub and C. F. Van Loan, *Matrix computations* (Johns Hopkins studies in the mathematical sciences), 3rd ed. Baltimore, MD, USA: Johns Hopkins University Press, 1996.

[69] D. Cox and D. Oakes, *Analysis of Survival Data* (Monographs on Statistics and Applied Probability). Washington, DC, USA: Chapman & Hall, 1984, vol. 21.

[70] A. Rohatgi, *WebPlotDigitizer*, version 4.5.

[71] M. Fakhruddin et al., "Hot carrier degradation and performance of 65nm RF n-MOSFET," in *2007 IEEE Radio Frequency Integrated Circuits (RFIC) Symp.*, Jun. 2007, pp. 551–554.

[72] J. S. Yuan, Y. Xu, S. D. Yen, Y. Bi, and G. W. Hwang, "Hot carrier injection stress effect on a 65 nm LNA at 70 GHz," *IEEE Trans. Device Mater. Reliab.*, vol. 14, no. 3, pp. 931–934, Sep. 2014.

[73] X. Xie, X. Zhang, W. Li, and X. Fan, "Hot carrier effect on NMOSFETs by stripe and annular gate design," in *2018 10th Int. Conf. Communications, Circuits Systems (ICCCAS)*, Dec. 2018, pp. 102–106.

[74] Z. Ren et al., "TID response and radiation-enhanced hot-carrier degradation in 65-nm nMOSFETs: Concerns on the layout-dependent effects," *IEEE Trans. Nucl. Sci.*, vol. 68, no. 8, pp. 1565–1570, Aug. 2021.

[75] *Virtex-5 FPGA Data Sheet: DC and Switching Characteristics*, DS202, Advanced Micro Devices, Inc., Santa Clara, CA, USA, 2016.

Appendix A

hci_param_tune.m

This MATLAB script performs steps 1,2,5, and 6 depicted in figure 5.1. It cannot be run in its present form as it is an amalgam of code from a MATLAB LiveScript and is incomplete. Data sourced from published studies is not included. A few lines must be written to import such data into a table (lines 17-25). The fitting process was performed manually and is configured for the specific data used in chapter 5. This could be improved by adding automation to edit the HSPICE netlist from appendix B, run HSPICE, and edit the python scripts that process the simulation results (appendices C and D). As currently written, the script must be suspended between fit attempts since each one generates simulation parameters, and depends upon the results of the previous simulation. Processed simulation results also must be loaded. This done on lines 13, 128, 149, 169, which must be edited.

```
1  %CONSTANTS
2  CONST.HOUR = 3600;
3  CONST.DAY = CONST.HOUR*24;
4  CONST.YEAR = CONST.DAY*365;
5  CONST.QTSMC = 6.75*CONST.YEAR; %TSMC defined 1E-3 failure quantile
6  CONST.k = 8.617e-5;%Boltzmann const in eV
7  CONST.T25 = 298; %25C
8  CONST.L65 = 0.065; %min gate length in um
9  CONST.L60 = 0.06; %actual min gate length in HSPICE in um
10 %default MOSRA HCI parameters
11 params_default = table(0,9.0,0,0.05,2.0,0.5,'VariableNames'...
12     ,{'Ap','D','G','Ea','m','n'});
13 %PATH_MC=path/to/MCDATA
14 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
15 %%PREPARE DATA
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%data_all: table containing raw data pulled from papers
%data loading and data_all construction not shown
%table variables are:
%deltaID: relative ID degradation, percent
%VDS: drain-source voltage,V
%VGS: gate-source voltage,V
%T: temperature, Kelvin
%L: gate length, um
%t: age, s

NDATA = size(data_all,1);
%deltaID to ln scale, remove T and L dependence
LDID_data = log(abs(data_all.deltaID));
%cancel out T influence
LDID_data = LDID_data + params_default.Ea./(CONST.k*data_all.T);
%cancel out L influence
LDID_data = LDID_data + params_default.m*data_all.L;
%Create stress matrix
S_data = [ones(size(LDID_data)) data_all.VDS data_all.VGS log(data_all.t)];

%table to hold synthetic data points (Lambda) used in parameter fits
%A new Lambda is created and appended to the table for each fit attempt
%create initial point
data_lambda = table(-10,1.1,1.1,CONST.T25,CONST.L60,CONST.QTSMC,...
    'VariableNames',{'deltaID','VDS','VGS','T','L','t'});
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%DETERMINE LAMBDA WEIGTH
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Lambda weights to try and equivalent number of copies of Lambda
lambda_w = [1 5 10:10:100]/100;
lambda_n = round(lambda_w*NDATA);
%sweep t_Lambda by +-10% of QTSMC
%t_Lambda is the failure time of Lambda
t_scale = 0.9:0.02:1.1;
%pre-generate a set of Lambda points each with a different t_Lambda
lambda_tscale = repmat(data_lambda,numel(t_scale),1);
lambda_tscale.t = lambda_tscale.t.*t_scale';
%convert Lambda degradation to ln scale, remove T and L influence like for real data
LDID_lambda = log(abs(lambda_tscale.deltaID(1)));
LDID_lambda = LDID_lambda + params_default.Ea./(CONST.k*lambda_tscale.T(1));
LDID_lambda = LDID_lambda + params_default.m*lambda_tscale.L(1);
%Stresss matrix for Lambda
S_lambda_tscale = [ones(size(lambda_tscale.deltaID))...
    lambda_tscale.VDS lambda_tscale.VGS log(lambda_tscale.t)];

%preallocate: store the weight, t_Lambda, t_fail,
% and solved parameters for each trial
lambda_sensitivity =...
    repmat(params_default,numel(lambda_n)*numel(t_scale),1);
nlambda = nan(numel(lambda_n)*numel(t_scale),1);
tlambda = nan(numel(lambda_n)*numel(t_scale),1);
tfail = nan(numel(lambda_n)*numel(t_scale),1);
lambda_sensitivity = addvars(lambda_sensitivity,nlambda...
    ,tlambda,tfail,'Before','Ap');
clear tfail tlambda nlambda;
```

```
72
73  %look for t_FAIL up to 10 years
74  tsim = linspace(0,10*CONST.YEAR);
75
76  rowind = 0;
77  for ind_ln = 1:numel(lambda_n)
78      NLAMBDA = lambda_n(ind_ln);
79      for ind_tscale = 1:numel(t_scale)
80          rowind = rowind+1;
81          LDID_sens = [LDID_data;repmat(LDID_lambda,NLAMBDA,1)];
82          S_sens = [S_data;repmat(S_lambda_tscale(ind_tscale,:),NLAMBDA,1)];
83          params_sens = S_sens\LDID_sens;
84          params_sens(1) = exp(params_sens(1));
85          lambda_sensitivity(rowind,{'nlambda','tlambda','Ap','D','G','n'})...
86              =[{NLAMBDA,t_scale(ind_tscale)},num2cell(params_sens')];
87          lambda_sensitivity(rowind,'tfail') =...
88              {mosraHCI_TSMC_tfail(lambda_sensitivity(rowind,4:9))};
89      end
90  end
91  %sensitivity slopes for each Lambda weight
92  tfail_all = reshape(lambda_sensitivity.tfail,numel(t_scale),[])/CONST.QTSMC;
93  lambda_ss = table(lambda_w(:),nan(numel(lambda_w),1),...
94      'VariableNames',{'weight','slope'});
95  for ind_slp = 1:numel(lambda_w)
96      p = polyfit(t_scale(:),tfail_all(:,ind_slp),1);
97      lambda_ss(ind_slp,'slope') = {p(1)};
98  end
99  %look for weight when slope starts to flatten out, eyeball it
100 figure
101 plot(lambda_ss.weight,lambda_ss.slope,'.-')
102 xlabel("\Lambda weight")
103 ylabel("Sensitivity Slope")
104 %use 40% weight
105 LW = 0.4;
106 NLAMBDA = lambda_n(lambda_w == LW);
107 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
108 %%MODEL PARAMS: v1
109 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
110 params_solve = params_default;
111
112 %store all the lambda points used, starting with the default
113 lambda = data_lambda;
114
115 LDID_lambda = log(abs(lambda.deltaID));
116 LDID_lambda = LDID_lambda + params_default.Ea/(CONST.k*lambda.T(1)) + params_default.m*lambda.L(1);
117 S_lambda = [1 lambda.VDS(1) lambda.VGS(1) log(lambda.t(1))];
118 LDID = [LDID_data;repmat(LDID_lambda,NLAMBDA,1)];
119 %stress matrix
120 S = [S_data;repmat(S_lambda,NLAMBDA,1)];
121
122 params_new = S\LDID;
123 params_solve = extendptable(params_solve,params_new);
124 disp(params_solve)
125 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
126 %%HSPICE RESULTS: v1
127 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
buff = readmatrix(fullfile(PATH_MC,'core_v1_fails_10percent.csv'));
mc_core(1) = pop_mc_result(buff(:,1));
fprintf('Q_0.001 for v1: %5.3f',mc_core(1).Q/CONST.YEAR)
%Q_0.001 is short, go again
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%MODEL PARAMS: v2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%adjust Lambda
delta_lambda_t = (CONST.QTSMC - mc_core(1).Q)/...
    lambda_ss.slope(lambda_ss.weight==LW);
lambda(2,:) = lambda(1,:);
lambda.t(2) = lambda.t(2) + delta_lambda_t;
%new parameter values
S_lambda = [1 lambda.VDS(2) lambda.VGS(2) log(lambda.t(2))];
S = [S_data;repmat(S_lambda,NLAMBDA,1)];
params_new = S\LDID;
params_solve = extendptable(params_solve,params_new);
disp(params_solve)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%HSPICE RESULTS: v2
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
buff = readmatrix(fullfile(PATH_MC,'core_v2_fails_10percent.csv'));
mc_core(2) = pop_mc_result(buff(:,1));
fprintf('Q_0.001 for v2: %5.3f',mc_core(2).Q/CONST.YEAR)
%Q_0.001 a bit long, go again
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%MODEL PARAMS: v3
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%adjust Lambda using HSPICE results
Qlin = polyfit([mc_core.Q],lambda.t(1:2)',1);
lambda(3,:) = lambda(2,:);
lambda.t(3) = polyval(Qlin,CONST.QTSMC);
%new parameter values
S_lambda = [1 lambda.VDS(3) lambda.VGS(3) log(lambda.t(3))];
S = [S_data;repmat(S_lambda,NLAMBDA,1)];
params_new = S\LDID;
params_solve = extendptable(params_solve,params_new);
disp(params_solve)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%HSPICE RESULTS: v3
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
buff = readmatrix(fullfile(PATH_MC,'core_v3_fails_10percent.csv'));
mc_core(3) = pop_mc_result(buff(:,1));
fprintf('Q_0.001 for v3: %5.3f',mc_core(3).Q/CONST.YEAR)
%Done!

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%HELPER FUNCTIONS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function t = mosraHCI_TSMC_tfail(mod_t)
    %find when a typical TSMC 65 nm core nMOSFET will fail
    %when stressed at VDS=VGS=1.1 V
    %mod_t: table of model parameter values
    k = 8.617e-5;%Boltzmann const in eV
    VDS = 1.1;
```

```
184     VGS = 1.1;
185     T = 298;
186     L = 0.06;
187     ln_dID = log(10);
188
189     ln_t = ln_dID -...
190         log(mod_t.Ap) -...
191         mod_t.D*VDS - mod_t.G*VGS +...
192         mod_t.Ea/(k*T) + mod_t.m*L;
193     ln_t = ln_t/mod_t.n;
194     t = exp(ln_t);
195 end
196 function ptable = extendptable(pt,np)
197     %append a new row of parameters to an existing parameter table
198     %pt: existing parameter table (table)
199     %np: new parameters (vector)
200     ptable = [pt;pt(end,:)];
201     ptable.Ap(end) = exp(np(1));
202     ptable.D(end) = np(2);
203     ptable.G(end) = np(3);
204     ptable.n(end) = np(4);
205 end
206 function mc = pop_mc_result(t_fail)
207     %populate Monte Carlo results from failure times
208     %return mc struct with fields
209     %%%%%mc.t_fail : failure times
210     %%%%%mc.F: ECDF values
211     %%%%%mc.t_ecdf: times for F
212     %%%%%mc.Q: 10^-3 quantile
213     %%%%%mc.Q_lo: lower 99% conf. bound on Q
214     %%%%%mc.Q_up: upper 99% conf. bound on Q
215     F_ref = 1e-3; %desired F value at Q
216     alph = 0.01; %conf. bound alpha value
217     mc.t_fail = t_fail;
218     [mc.F,mc.t_ecdf,Flo,Fup] = ecdf(t_fail,'Alpha',alph);
219     %find the ECDF quantile closest to F_ref
220     F_i(1) = find(mc.F<=F_ref,1,'last');
221     F_i(2) = find(mc.F>=F_ref,1);
222     [~,Q_i] = min(abs(mc.F(F_i) - F_ref));
223     mc.Q = mc.t_ecdf(F_i(Q_i));
224     %do the above again for confidence bounds
225     %Q lower bound, use upper bound of F since it occurs first
226     blo_i(1) = find(Fup<=F_ref,1,'last');
227     blo_i(2) = find(Fup>=F_ref,1);
228     [~,blomin] = min(abs(Fup(blo_i) - F_ref));
229     mc.Q_lo = mc.t_ecdf(blo_i(blomin));
230     %Q upper bound, use lower bound of F since it occurs last
231     bup_i(1) = find(Flo<=F_ref,1,'last');
232     bup_i(2) = find(Flo>=F_ref,1);
233     [~,bupmin] = min(abs(Flo(blo_i) - F_ref));
234     mc.Q_up = mc.t_ecdf(bup_i(bupmin));
235 end
```

Appendix B

hci_core_mc.sp

This HSPICE netlist file ages a minimum sized core nMOSFET under accelerated HSPICE stress. It is configured to perform this as a Monte Carlo simulation that ages `p_monte` devices. It is used to perform step 3 depicted in figure 5.1. A `radeg` file containing nMOSFET degradation values is generated for each Monte Carlo instance. No other output is generated as only the pre-stress simulation is performed.

```
1  HCI CORE NMOS
2  *hci_core_mc.sp
3  *Monte Carlo MOSRA
4  *HCI aging of a core nmos (CRN65GP)
5  ****************
6  *PARAMETERS
7  **number of fingers: 1-32
8  .param nfing = 1
9  **stressing voltages
10 .param stressvd = 1.1
11 .param stressvg = 1.1
12 **MOSRA aging settings
13 .param p_reltotaltime = 8yr
14 .param p_relstep = 0.005yr
15 **Monte Carlo
16 .param p_monte = 10000 $MC sample size
17 ****************
18 *LIBRARY INCLUDE
19 **glogal variations
20 .lib '/CMC/kits/tsmc_65nm/CRN65GP/SPICE/crn65gplus_2d5_lk_v1d1_usage.l' mc_lib
21 **local variations only
22 *.lib '/CMC/kits/tsmc_65nm/CRN65GP/SPICE/crn65gplus_2d5_lk_v1d1_usage.l' mismatch_lib
23 **typical models
24 *.lib '/CMC/kits/tsmc_65nm/CRN65GP/SPICE/crn65gplus_2d5_lk_v1d1_usage.l' tt_lib
25 ****************
26 *OPTIONS
```

```
**print extra circuit info in output listing file
*.option list $circuit element summary and parameter defns
*.option node $ist of nodes and elements connected to each node
*--
**new listing style: sends print output to separate file
*For MC keep off, not compatible with mramcfile
*.option lis_new=1
**simplify output listing
.option nomod=1 $suppress circuit model parameter details
.option lislvl=1 $suppress circuit number to circuit hierarchy
*--
**MOSRA options
.option appendall $REQUIRED for mosfet inside a subckt
.option radegoutput=YES $create degradation file
****************
*ANALYSIS
.temp 25
.tran 1n 1n sweep monte=p_monte
**aging analysis
.mosra reltotaltime=p_reltotaltime relstep=p_relstep relmode=1 simmode=0
****************
*NETLIST
vds d 0 stressvd
vgs g 0 stressvg
xn d g 0 0 nmos_rf lr=0.06u wr=1.0u nr=nfing

****************
*MOSRA
*define MOSRA model for HCI
.model hci_mosra mosra level=3
+HCIAP=-1.1271E-04 HCIG=2.2461E+00 HCID=4.1104E+00 HCIN=3.3726E-01
.mosra_subckt_pin_volt xn $use subckt pin voltages instead of mosfet terminals for MOSRA
.appendmodel hci_mosra mosra "*" nmos $append MOSRA model to all nMOS model cards
.end
```

Appendix C

radeg_pull.py

Collate drain current degradation data from a set of HSPICE `radeg` files located in a common directory. Simulation results for circuits with an arbitrary number of MOSFETs can be processed. Output to both MATLAB and NumPy files is possible. The saved data is a 3-dimensional matrix with the following indexing:

1. Monte Carlo index
2. Device index
3. Age index

Invocation: **radeg_pull.py** [**-h**] *inputdir outfile* **-f m|n|b**

- -h: help message
- inputdir: directory containing radeg files to be processed
- outfile: output file name, will be saved in current directory
- -f: output file format, m: MATLAB, n: NumPy, b: both

```
#!/home/adimopou/anaconda3/bin/python

import argparse
from pathlib import Path
import re
import numpy as np
import scipy.io

def get_infile_paths(indir_path: Path, valid_pattern: str) -> Path.glob:
    """
----Validate and return paths to input data files.

----Raises an exception if:
----1) indir_path is not a valid path
----2) no files match valid_pattern

----Parameters
--------------
----indir_path
--------Path to directory containing the data files.
----valid_pattern
--------Glob pattern specifying valid data file names.

----Returns
----------
----Path.glob generator for valid input data files
----"""
    if not indir_path.is_dir():
        raise FileNotFoundError('ERROR: '+str(indir_path)+' is not a directory')

    infile_paths = indir_path.glob(valid_pattern)

    if not any(infile_paths):
        raise FileNotFoundError('ERROR: no valid data files in '+str(indir_path))

    #recreate infile_paths because any() consumes first element
    #I don't know how to recover it
    infile_paths = indir_path.glob(valid_pattern)
    return infile_paths

def measure_record(record_file_path: Path):
    """
----Determine line and character indices of relevant information in a record.

----Parameters
----------------
----record_file_path
--------Path of record file

----Returns
----------
----Dictionary with following keys:
--------n_neaders: number of header lines in file
--------rec_len: length of a record in lines
--------dev_names: list of the names of aged devices
--------age_start_ind: first character index of the circuit age value
```

```
57  --------deg_lines:-line-numbers-of-degradation-values,-relative-to-start-of-a-record
58  --------deg_start_ind:-first-character-index-of-the-device-degradation-value
59  ----"""
60      rec_start_tag = "Circuit time"
61      deg_tag = "dids"
62      rec_bounds = []
63      dev_rec_starts = []
64      dev_names = []
65      age_start_ind = 0
66      deg_start_ind = 0
67      deg_lines = []
68      return_dict = {}
69      with open(record_file_path,'r') as rec_file:
70          for line_num,line in enumerate(rec_file):
71              #find start/end lines of record
72              if line.strip().startswith(rec_start_tag):
73                  rec_bounds.append(line_num)
74                  if len(rec_bounds) > 1:
75                      break
76                  age_start_ind = re.search(r"\d+\.?\d*",line).start()
77              #devices separated by a blank line, make sure to have passed header
78              if rec_bounds and (not line.strip()):
79                  dev_rec_starts.append(line_num+1)
80              if line.strip().startswith(deg_tag):
81                  deg_lines.append(line_num - rec_bounds[0])
82                  #start of deg value on line, only do this once
83                  if not deg_start_ind:
84                      deg_start_ind = re.search(r"\d+\.?\d*",line).start()
85          dev_rec_starts.pop()#record ends with empty line
86          rec_file.seek(0,0)
87          #now get the device names
88          for line_num,line in enumerate(rec_file):
89              if line_num in dev_rec_starts: dev_names.append(line.strip())
90
91      return_dict.update([("n_headers",rec_bounds[0])])
92      return_dict.update([("rec_len",rec_bounds[1]-rec_bounds[0])])
93      return_dict.update([("dev_names",dev_names)])
94      return_dict.update([("age_start_ind",age_start_ind)])
95      return_dict.update([("deg_lines",deg_lines)])
96      return_dict.update([("deg_start_ind",deg_start_ind)])
97      return return_dict
98
99  def extract_age_data(infile_paths: Path.glob):
100     """
101 ----Extract-the-following-values-from-a-set-of-record-files:
102 ----circuit-age
103 ----names-of-aged-devices
104 ----degradations-of-each-device
105
106 ----Parameters
107 --------------
108 ----infile_paths
109 --------Path-iterable-for-valid-data-files
110
111 ----Returns
112 ----------
```

```
113 ----Dictionary with the following keys:
114 --------age: circuit ages in s
115 --------dev_names: name of each aged device
116 --------dids: Delta I_D for each device at each age, across entire Monte Carlo ensemble
117 ------------Dimensions are
118 ----------------0: Monte Carlo index
119 ----------------1: device index
120 ----------------2: age index
121 ----"""
122     rec_measured = False
123     first_file = True
124     dids_ensemble = []
125 
126     for fp in(infile_paths):
127         if not rec_measured:
128             rec_meta = measure_record(fp)
129             rec_measured = True
130         dids = []
131         with open(fp,'r') as rec_file:
132             #skip past the header
133             for hline in range(rec_meta["n_headers"]):
134                 next(rec_file)
135             rec_lines = rec_file.readlines()
136         if first_file:
137             ages = [a[rec_meta["age_start_ind"]:].strip() for a in rec_lines[0::rec_meta["rec_len"]]]
138             first_file = False
139         for dev_ind in range(len(rec_meta["deg_lines"])):
140             dids.append([d[rec_meta["deg_start_ind"]:-2].strip() for d in rec_lines[rec_meta["deg_lines"][
                    dev_ind]::rec_meta["rec_len"]]])
141         dids_ensemble.append(dids)
142     return dict([("age",np.array(ages,dtype=float)),("dids",np.array(dids_ensemble,dtype=float)),("
            dev_names",rec_meta["dev_names"])])
143 
144 
145 ################################################
146 ################################################
147 #BEGIN SCRIPT
148 
149 #parse CLI
150 parser = argparse.ArgumentParser(description='Collate all dids values generated by a MOSRA simulation. 
        Also works in Monte Carlo case.')
151 parser.add_argument('inputdir',help='Directory containing simulation radeg files.',type=Path)
152 parser.add_argument('outfile',help='Output file name. Do not provide an extension.',type=Path)
153 parser.add_argument('-f',type=str,choices=['m','n','b'],default='b',help='Output format. Options are m for
        MATLAB, n for NumPy, b for both (default).')
154 args = parser.parse_args()
155 
156 #get generator for valid data file paths
157 #valid data files have radeg0 in their names
158 try:
159     infiles = get_infile_paths(args.inputdir,'*radeg0*')
160 except FileNotFoundError as err:
161     print(err)
162     raise SystemExit
163 
164 data = extract_age_data(infiles)
```

```
165 #save the result
166 if args.f == 'n' or args.f == 'b':
167     np.savez_compressed(args.outfile, age=data["age"],dids=data["dids"],dev_names=data["dev_names"])
168 if args.f == 'm' or args.f == 'b':
169     scipy.io.savemat(args.outfile.with_suffix('.mat'),data)
```

Appendix D

mc_fail_stats.py

Compute the empirical cumulative distribution function (ECDF) for MOSFET failure times. Used to perform step 4 depicted in figure 5.1.

Invocation: **mc_fail_stats.py** *degdatafile.npz FAILP*

- degdatafile.npz: input NumPy file generated by radeg_pull.py (appendix C)
- FAILP: percentage degradation in drain current to consider as the threshold for a MOSFET failure, range is [0,100]

Two CSV files are generated:

- ecdf_*FAILP*percent.csv: the computed ECDF
- fails_*FAILP*percent.csv: the raw failure times used to compute the ECDF

```
1  #!/home/adimopou/anaconda3/bin/python
2
3
4  import sys
5  import numpy as np
6
7
8  def find_fail_times(data, fail_threshold):
9      """find age when a device has failed
10
11 ----Args:
12 --------data (numpy array (n,2,m)): degradation data sets, generated by pull_age_deg()
13 --------fail_threshold (float): Id relative to age 0. E.g. 90.0 == 90% of fresh performance
```

```
14
15  ----Returns:
16  --------NumPy array of shape (n_sets,2), type float.
17  --------Column 0 is failure time (s).
18  --------Column 1 is actual degradation at failure time (% delta Id)
19  --------NaN indicates that failure did not occur (censored data)
20  ----"""
21      #preallocate return array with censored values, then populate
22      fail_data=np.nan*np.ones((data.shape[0],2),dtype=float)
23
24      #find earlierst failure in each data set
25      for set_index,set_data in enumerate(data):
26
27          #exception is thrown if a failure cannot be found in data_set
28          try:
29              #degradation data is monotonic in time so data_set is already sorted
30              #searchsorted is a bit faster than argwhere, even when flipping the sign
31              #so that data is in ascending rather than descending order
32              fail_index = np.searchsorted(-set_data[1],-fail_threshold)
33              fail_data[set_index,:] = set_data[:,fail_index].ravel()
34          except:
35              #failure not found in data_set
36              #fail_data pre-populated with NaN, no need to act here
37              pass
38
39      return fail_data
40
41  def ecdf(failures):
42      """Empirical cummulative distribution function based on Kaplan-Meier estimator.
43  -------Accounts for right-censored data only (failure occurs after end of recorded data).
44
45  ----Args:
46  --------failures (numpy array (n,1)): failure times, censored data = np.nan
47
48  ----Returns:
49  --------NumPy array, shape (n,2)
50  --------ecdf[0] = time (s)
51  --------ecdf[1] = cummulative distribution
52  ----"""
53      #total units initially at risk, including censored
54      n_total = len(failures)
55
56      #number of failures during each time interval
57      #censored values get stacked at the end
58      [fail_time,fail_count] = np.unique(failures,return_counts=True)
59      #remove censored data if present
60      if np.isnan(fail_time[-1]):
61          censored_start = np.argwhere(np.isnan(fail_time)).item(0)
62          fail_time = fail_time[:censored_start]
63          fail_count = fail_count[:censored_start]
64
65      #add a data point for t=0, represents time interval [0,fail_time[0])
66      fail_count = np.insert(fail_count,0,0.0)
67      fail_time = np.insert(fail_time,0,0.0)
68
69      #count number of units at risk of failure at start of each time interval
```

```
70      n_atrisk = np.zeros(fail_count.shape)
71      n_atrisk[0] = n_total
72      for risk_index in range(1,len(n_atrisk)):
73          n_atrisk[risk_index] = n_atrisk[risk_index-1] - fail_count[risk_index-1]
74
75      #survival function
76      # S(t) = product(1 - fail_count[i]/n_atrisk[i]) for for i=0...t
77      #this reduces to S(t) = n_atrisk[t+1]/n_total
78      survival = np.zeros(np.size(n_atrisk))
79      survival[:-1] = n_atrisk[1:]/n_total
80      #need to create an extra element of n_atrisk on the fly for the last time point
81      survival[-1] = (n_atrisk[-1] - fail_count[-1])/n_total
82
83      return np.column_stack((fail_time,1-survival))
84
85  ###================================================================
86  ### SCRIPT START
87  #get input file names
88  #This script works on data files returned by mc_did_pull.py
89  input_f_name = sys.argv[1]
90  deg_percent = np.float(sys.argv[2])
91
92  loaded_data = np.load(input_f_name)
93  raw_data = loaded_data['arr_0']
94  loaded_data.close()
95
96  f_data = find_fail_times(raw_data,100.0 - deg_percent)
97  dist = ecdf(f_data[:,0])
98
99  #save result in CSV file
100 #make file names based on deg_percent
101 #replace decimal point with _ if needed
102 if (np.ceil(deg_percent)-deg_percent) != 0:
103     f_out_tag = str(deg_percent)
104     f_out_tag = f_out_tag.replace('.','_')
105 else:
106     f_out_tag = str(int(deg_percent))
107 f_out_tag = f_out_tag + 'percent.csv'
108
109 np.savetxt('ecdf_'+f_out_tag,dist,fmt='%1.6e,%1.6e',header='age-(s),-cummumaltive-failure')
110 np.savetxt('fails_'+f_out_tag,f_data,fmt='%1.6e,%1.6e',header='age-(s),-delta-ID-(%)')
```

Appendix E

mc_readmode.sh

A shell script for automating Monte Carlo simulations of the anti-recycling sensor described in chapter 6. Relevant lines in tharcdc.sp (appendix F) are edited using sed before the netlist is run through HSPICE. Simulation results are extracted and collated using radeg_pull.py (appendix C), hspice_RAN_pull.py (appendix G), and hspice_collate_MC_RAN.py (appendix H). Invocation: **mc_readmode.sh** *netlist stressT stressVDD*

- netlist: HSPICE netlist for the circuit to be simulated
- stressT: simulation temperature to be used in the stress-mode pre-simulation
- stressVDD: simulation VDD to be used in the stress-mode pre-simulation

All read-mode post-simulations use the same conditions: temperature=25 °C, VDD=1.8 V. Stress-mode simulation radeg files are stored in a locally-created subdirectory called radeg. Read-mode simulation results are collated and stored as a MATLAB file in a locally-created subdirectory called data_dir.

```
1  #!/usr/bin/sh
2
3  if [[ $# -lt 1 ]]; then
4      echo "missing argument:"
5      echo "spice file"
6      exit 1
7  fi
8
```

```
9  if [[ $# -lt 2 ]]; then
10     echo "missing argument:"
11     echo "stress temperature"
12     exit 1
13 fi
14
15 if [[ $# -lt 3 ]]; then
16     echo "missing argument:"
17     echo "stress VDD"
18     exit 1
19 fi
20
21 spfile=$1
22 stress_temp=$2
23 stress_vdd=$3
24 read_temp=25
25 read_vdd=1.8
26 radeg_dir=radeg
27 data_dir=data_out
28 datafile_name=tharcdc_readmode_T${stress_temp}_VDD$(echo "scale=0;$stress_vdd*1000/1" |bc)
29 radegfile_name=${datafile_name}_radeg
30
31 if [[ !(-f $spfile && $spfile =~ \.sp$) ]];then
32     echo "File $spfile does not exist or is not a SPICE file."
33     exit 1
34 fi
35 if [[ !(-d $radeg_dir) ]];then
36     echo "**** Creating directory: $radeg_dir"
37     mkdir radeg
38 fi
39 if [[ !(-d data_dir) ]];then
40     echo "**** Creating directory: $data_dir"
41     mkdir $data_dir
42 fi
43 ##############################################
44 echo -e "\n----------------------------------------"
45 echo "MONTE CARLO MOSRA PRE-STRESS"
46 echo "----------------------------------------"
47 sed -i "/\.param mosrasimmode/ s/[0-9]/0/" $spfile
48 sed -i "/\.param anselect/ s/[0-9]/0/" $spfile
49 sed -i "/\.param sensormodeselect/ s/[0-9]/0/" $spfile
50 sed -i "/\.temp/ s/[0-9][0-9]*/$stress_temp/" $spfile
51 sed -i "/\.param VDD_IO_VOLT/ s/[0-9].[0-9]*/$stress_vdd/" $spfile
52 hspice -mt 2 -i $spfile -o sm0
53 mv *.radeg* $radeg_dir
54 ##############################################
55 echo -e "\n----------------------------------------"
56 echo "MONTE CARLO MOSRA POST-STRESS"
57 echo "----------------------------------------"
58 sed -i "/\.param mosrasimmode/ s/[0-9]/1/" $spfile
59 sed -i "/\.param anselect/ s/[0-9]/1/" $spfile
60 sed -i "/\.param sensormodeselect/ s/[0-9]/1/" $spfile
61 sed -i "/\.temp/ s/[0-9][0-9]*/$read_temp/" $spfile
62 sed -i "/\.param VDD_IO_VOLT/ s/[0-9].[0-9]*/$read_vdd/" $spfile
63 radeg_files=($radeg_dir/*.radeg*)
64 radeg_files_sorted=($(echo "${radeg_files[@]}" | tr ' ' '\n' | sort -V))
```

```
65  for f_ind in "${!radeg_files_sorted[@]}"
66  do
67      mc_ind=$(($f_ind+1))
68      rf="${radeg_files_sorted[$f_ind]}"
69      sed -i "/\.param sm1_mcind/ s/[0-9][0-9]*/$mc_ind/2" $spfile
70      #use | for sed delimiter because / is in $rf string
71      #begin with escape \| to specify that | is the delimiter
72      sed -i "\|\.option radegfile| s|'.*'|'$rf'|" $spfile
73      echo "****MONTE-CARLO-INDEX: $mc_ind"
74      hspice -mt 2 -i $spfile -o sm1@$mc_ind
75  done
76  ##############################################
77  echo -e "\n----------------------------------------"
78  echo "DATA-EXTRACTION"
79  echo "----------------------------------------"
80  echo "****RADEG"
81  radeg_pull.py $radeg_dir $radegfile_name -f m
82  echo "****extract measurement data"
83  lis_files=(sm1*.lis)
84  for lf in "${lis_files[@]}"
85  do
86      hspice_RAN_pull.py $lf -f n
87  done
88  echo "****collate measurement data"
89  tmpdatafiles=(sm1*.npz)
90  tmpdatafiles_sorted=($(echo "${tmpdatafiles[@]}" | tr ' ' '\n' | sort -V))
91  hspice_collate_MC_RAN.py "${tmpdatafiles_sorted[@]}" $datafile_name -f m
92  echo "****cleanup"
93  rm -v "${tmpdatafiles_sorted[@]}"
94  mv *mat $data_dir
95  rm -v sm0*
96  rm -v sm1*
97  rm -v ${radeg_dir}/*
98  ##############################################
99  echo -e "\n----------------------------------------"
100 echo "COMPLETE"
101 echo "----------------------------------------"
```

Appendix F

tharcdc.sp

HSPICE netlist for the anti-recycling sensor described in chapter 6. Simulation settings are controlled by the user-set parameters in the SIM CTRL OPTS section (lines 15-32). As discussed in section 4.2, HSPICE will not properly run a post-stress Monte Carlo simulation with MOSRA. The workaround requires iterating over the radeg file previously generated, specifying the current Monte Carlo index, and running a Monte Carlo simulation with a single instance. This is only feasible through automation by a script such as the one in appendix E.

```
1  THARCDC
2  *thardc.sp
3  *====================
4  *====DESCRIPTION
5  *====================
6  *Aged element is a min sized nMOS with:
7  *G set by V divider
8  *D set by pMOS pullup
9  *G&D voltage close to equal initially
10 *D tied to output path for measurement through pass gate.
11 *I/O ESD protection by GGNMOS: lower leakage and more linear than diodes
12 *separate pre-stress and post-stress sims possible
13 *pre-stress: TRAN, circuit in stress mode
14 *post-stress: DC, circuit in either stress or read mode
15 *====================
16 *====SIM CTRL OPTS
17 *====================
18 **LIBRARY SELECT
19 *0: TT (typical)
20 *1: MC (monte carlo, global var only)
21 *2: mismatch (monte carlo, local var only)
22 .param libselect=1
```

```
**MOSRA
.param mosraselect=0 $----1:perform MOSRA, 0:no MOSRA
**MOSRA simmode
.param mosrasimmode=2 $----0:pre-stress simulation,1:post-stress simulation,2:both
**analysis selection
.param anselect=1 $----0:TRAN, 1:DC
**select constant or dynamic transient
.param trantypeselect=0 $----0:constant voltages, 1:dynamic voltage waveforms
**stressing or reading the aging element
.param sensormodeselect=1 $----0:stress, 1:read

*====================
*====LIBRARY
*====================
.if (libselect == 0)
    *Typical
    .lib '/CMC/kits/tsmc_65nm/CRN65GP/SPICE/crn65gplus_2d5_lk_v1d1_usage.l' tt_lib
.elseif (libselect == 1)
    *Global variations
    .lib '/CMC/kits/tsmc_65nm/CRN65GP/SPICE/crn65gplus_2d5_lk_v1d1_usage.l' mc_lib
.elseif (libselect == 2)
    *Local variations only
    .lib '/CMC/kits/tsmc_65nm/CRN65GP/SPICE/crn65gplus_2d5_lk_v1d1_usage.l' mismatch_lib
.endif
*====================
*====PARAMETERS
*====================
**GEOMETRY
.param scale_mos_rf=0.06u $TSMC REQUIRED for scaling, core
.param scale_mos_rf_18=0.06u $TSMC REQUIRED for scaling, I/O
.param W_APG=39
.param W_RPG=166
**VOLTAGES
.param VDD_IO_VOLT=1.89
**TIMING
.if (anselect == 0)
    .param TSTEP=10n
    .if (trantypeselect == 0)
        .param TSTOP='5*TSTEP'
        .param TAGE=TSTOP
    .elseif (trantypeselect == 1)
        .param TAGE='10*TSTEP'
        .param TLH='10*TSTEP'
        .param TSETUP='10*TSTEP'
        .param TREAD='18*TSTEP'
        .param TSTOP='TAGE+TLH+TSETUP+TREAD'
    .endif
.endif
**DC ANALYSIS
.if (anselect == 1)
    .param VSTART=0
    .param VSTOP=VDD_IO_VOLT
    .param VSTEP=0.1
    .endif
**MOSRA
.if (mosraselect == 1)
```

```
79      .param agetotal=10*365*24*3600
80      .param agestep=5*24*3600
81  .endif
82  **MONTE CARLO
83  .if (libselect == 1 || libselect == 2)
84      .param nsamp=1000
85      .param sm1_mcind=1000
86  .endif
87  *====================
88  *====NETLIST
89  *====================
90  vddio vddio GND VDD_IO_VOLT
91  .if ((anselect == 0 && trantypeselect == 0) || anselect == 1)
92      .if (sensormodeselect == 0)
93          vin vin GND 0
94          vaen a_en_b GND 0
95          vren r_en GND 0
96          vrenn r_en_b GND VDD_IO_VOLT
97      .elseif (sensormodeselect == 1)
98          vin vin GND VDD_IO_VOLT
99          vaen a_en_b GND VDD_IO_VOLT
100         vren r_en GND VDD_IO_VOLT
101         vrenn r_en_b GND 0
102     .endif
103 .elseif (anselect == 0 && trantypeselect == 1)
104     vin vin GND PWL 0 0, 'TAGE+TLH+TSETUP' 0, TSTOP VDD_IO_VOLT
105     vaen a_en_b GND PWL 0 0, TAGE 0, 'TAGE+TLH' VDD_IO_VOLT, TSTOP VDD_IO_VOLT
106     vren r_en GND PWL 0 0, TAGE 0, 'TAGE+TLH' VDD_IO_VOLT, TSTOP VDD_IO_VOLT
107     vrenb r_en_b GND PWL 0 VDD_IO_VOLT, TAGE VDD_IO_VOLT, 'TAGE+TLH' 0, TSTOP 0
108 .endif
109
110 *v divider for xn_age gate
111 xp_div g_age g_age vddio vddio pmos_rf_18 lr=5 wr=17 nr=10
112 xn_div g_age g_age GND GND nmos_rf_18 lr=5 wr=17 nr=1
113 *aging pass gate, drain of xn_age
114 xp_apg d_age a_en_b vddio vddio pmos_rf_18 lr=5 wr=W_APG nr=2
115 *aging element
116 xn_age d_age g_age GND GND nmos_rf lr=1 wr=10 nr=1
117 *reading transmission gate
118 xn_rpg rnode r_en d_age GND nmos_rf_18 lr=5 wr=W_RPG nr=1
119 xp_rpg d_age r_en_b rnode vddio pmos_rf_18 lr=5 wr=W_RPG nr=3
120 *I/O protection block
121 xp_esd1 vin vddio vddio vddio pmos_rf_18 lr=5 wr=166 nr=4
122 xn_esd2 vin GND GND GND nmos_rf_18 lr=5 wr=166 nr=2
123 Resd vin rnode 200 $min R set by DRC for ESD
124 xp_esd3 rnode vddio vddio vddio pmos_rf_18 lr=5 wr=166 nr=4
125 xn_esd4 rnode GND GND GND nmos_rf_18 lr=5 wr=166 nr=2
126 *====================
127 *====OPTIONS
128 *====================
129 *--Scaling of primitive models
130 .option scale=0.06u
131 *--Tran options for higher accuracy
132 .option runlvl=6
133 .option kcltest
134 .option delmax=500p
```

```
**shorten lis file
.option nomod=1 $suppress circuit model parameter details
.option lislvl=1 $suppress circuit number to circuit hierarchy
.option opfile=1 $write operating point to separate file
**MOSRA
.if (mosraselect == 1)
    .option appendall $REQUIRED for mosfet inside a subckt
    **MOSRA with MC
    .if (libselect == 1 || libselect == 2)
        .option mramcfile=1$provide output for each relstep during monte carlo for post-stress
        .if (mosrasimmode == 1)
            .option radegfile='radeg/sm0.radeg0@1000'
        .endif
    .endif
.elseif (mosraselect == 0)
    **non-MOSRA print to separate file
    .option lis_new
.endif
*====================
*====ANALYSIS
*====================
.temp 75
.if (libselect == 0)
    .if (anselect == 0)
        .tran TSTEP TSTOP
    .elseif (anselect == 1)
        .dc vin VSTART VSTOP VSTEP
    .endif
.elseif (libselect == 1 || libselect == 2)
    .if (anselect == 0)
        .if (mosrasimmode == 0 || mosrasimmode == 2)
            .tran TSTEP TSTOP sweep monte=nsamp
        .elseif (mosrasimmode == 1)
            .tran TSTEP TSTOP sweep monte=1 firstrun=sm1_mcind
        .endif
    .elseif (anselect == 1)
        .if (mosrasimmode == 0 || mosrasimmode == 2)
            .dc vin VSTART VSTOP VSTEP sweep monte=nsamp
        .elseif (mosrasimmode == 1)
            .dc vin VSTART VSTOP VSTEP sweep monte=1 firstrun=sm1_mcind
        .endif
    .endif
.endif
**MOSRA
.if (mosraselect == 1)
    *define MOSRA model for HCI
    .model hci_core mosra level=3
    +HCIAP=-3.3191E-04 HCIG=2.0473E+00 HCID=4.8411E+00 HCIN=2.5063E-01
    .appendmodel hci_core mosra nch_rf nmos $append HCI model to core NMOS models
    .model hci_IO mosra level=3
    +HCIAP=-1.4171E-03 HCIG=0 HCID=3.5089E+00 HCIN=2.5805E-01
    .appendmodel hci_IO mosra nch_rf_18 nmos $append HCI model to IO NMOS models
    .mosra_subckt_pin_volt nmos_rf, nmos_rf_18 $use macromodel pin voltages, not MOSFET model pin voltages
    .if (anselect == 0)
        .mosra reltotaltime=agetotal relstep=agestep agingstop=TAGE relmode=1 simmode=mosrasimmode
    .elseif (anselect == 1)
```

```
191         .mosra reltotaltime=agetotal relstep=agestep relmode=1 simmode=mosrasimmode
192     .endif
193 .endif
194 *====================
195 *====OUTPUT
196 *====================
197 .if (anselect == 0)
198     .print tran v(vin) i(vin) v(rnode) v(d_age) v(g_age) v(a_en_b) v(r_en) v(r_en_b) i(xn_age.m0) i(xn_div.m0)
199 .elseif (anselect == 1)
200     .print dc v(vin) i(vin) v(rnode) v(d_age) v(g_age) v(a_en_b) v(r_en) v(r_en_b) i(xn_age.m0)
201 .endif
202 .END
```

Appendix G

hspice_RAN_pull.py

Extract HSPICE Reliability Analysis results printed to a listing file. Works with TRAN and DC analysis results, with or without Monte Carlo. DC analysis results must be post-simulation results.

Invocation: **hspice_RAN_pull.py** [**-h**] *lisfile* **-o** *outfile* **-f m|n|b**

- -h: help message
- lisfile: HSPICE listing file containing analysis results to be extracted
- outfile: output file name
- -f: output file format, m: MATLAB, n: NumPy, b: both

Saved variables are

- sweep: independent sweep variable (vector)
- sweep_hdr: header string for sweep (string)
- age: MOSRA age (vector)
- meas: analysis results (4-D array with indexing 1. variable index 2. sweep index 3. age index 4. Monte Carlo instance)

- meas_hdr: header string for meas containing analysis variable names (vector of strings)

```
#!/home/adimopou/anaconda3/bin/python
import argparse
from pathlib import Path
import numpy as np
import scipy.io
import re

def read_record(lisfile):
    """
----Generator. Iterate over all records in lisfile.

----Params:
----------
----lisfile: file object for lisfile containing TRAN printouts.

----Returns:
----------
----List of strings defining a complete TRAN record.
----"""
    in_block = False
    block = []
    for lis_line in lisfile:
        if 'begin' in lis_line:
            in_block = True
        if 'end' in lis_line:
            block.append(lis_line.strip())
            yield block
            in_block = False
            block = []
        if in_block and (not lis_line.isspace()):
            block.append(lis_line.strip())

def get_rec_profile(rec):
    """
----Identify important line numbers in the record

----Params:
----------
----rec: list
--------List of strings, each a line from the record.
--------Each line has been left and right stripped of white space.

----Returns:
----------
----Dict with following fields:
--------mcind: list of Monte Carlo index line numbers
--------datastarts: list of tuples, starts of data fields
--------dataends: list of tuples, ends of data fields
--------age: list length 1 with line number for MOSRA age of record
--------headers: list of tuples, header line numbers

----Each tuple in datastarts and dataends corresponds to a MC instance in the record
----"""
    ret_dict = {'mcind':[], 'datastarts':[], 'dataends':[], 'age':[], 'headers':[]}
```

```
55      #find every line containing monte carlo index
56      line_is_target = [bool(re.search(r'monte\s+carlo\s+index', l)) for l in rec]
57      ret_dict['mcind'] = [ind for ind,val in enumerate(line_is_target) if val]
58      #find start and end of all data series relative to a monte carlo index line
59      #there can be multiple data series if more than 4 variables were recorded
60      #because HSPICE has fixed width output
61      mc_inst_start_l = ret_dict['mcind']
62      mc_inst_end_l = []
63      if len(mc_inst_start_l) > 1:
64          mc_inst_end_l.extend(mc_inst_start_l[1:])
65      #cover non Monte Carlo case
66      if not ret_dict['mcind']:
67          mc_inst_start_l = [1]
68      mc_inst_end_l.append(-1)
69      for strt,stp in zip(mc_inst_start_l,mc_inst_end_l):
70          line_is_target = [l == 'x' for l in rec[strt:stp]]
71          ret_dict['datastarts'].append(tuple(ind+strt for ind,val in enumerate(line_is_target) if val))
72          line_is_target = [l == 'y' for l in rec[strt:stp]]
73          ret_dict['dataends'].append(tuple(ind+strt for ind,val in enumerate(line_is_target) if val))
74      #find MOSRA age line
75      line_is_target = [bool(re.search(r'reliability\s+time',l)) for l in rec]
76      ret_dict['age'] = [ind for ind,val in enumerate(line_is_target) if val]
77      #find header lines, will be multiple sets if >4 variables recorded
78      #headers usually at least 2 lines long, be ready for less or more lines just in case
79      for strt,stp in zip(ret_dict['datastarts'][0],ret_dict['dataends'][0]):
80          line_is_target = [bool(re.search(r'^[^\d]',l)) for l in rec[strt+1:stp]]
81          ret_dict['headers'].append(tuple(1+ind+strt for ind,val in enumerate(line_is_target) if val))
82      #advance datastarts past the header lines
83      header_len = ret_dict['headers'][0][-1] - ret_dict['datastarts'][0][0]
84      ret_dict['datastarts'] = [tuple([n+header_len+1 for n in m]) for m in ret_dict['datastarts']]
85      return ret_dict
86
87  def get_headers(raw_rec,rec_profile):
88      """
89  ----Extract headers from listing file.
90  ----Each variable has a header. All headers except
91  ----for the sweep variable span at least two lines.
92  ----Header lines for each variable flattened into a comma separated string.
93
94  ----Params:
95  ----------
96  ----raw_rec: raw record returned by read_record
97  ----rec_profile: dict returned by get_rec_profile
98
99  ----Returns:
100 ----------
101 ----List: header string for each variable
102 ----"""
103     #listing file has a fixed width so header lines may be discontinuous across multiple file lines
104     #regroup header lines to account for this e.g. header line 1 may be on file lines 1 and 23, make this
            one group
105     headers_group_ln = list(zip(*rec_profile['headers']))
106     hlines = []
107     #split header lines into columns, concatenate all columns from same group
108     for hgi,hg in enumerate(headers_group_ln):
109         hline_single = []
```

```
110         for li,ln in enumerate(hg):
111             cols = raw_rec[ln].split()
112             #sweep variable has a single header line,
113             #pad its column to keep constant width
114             if hgi > 0 and li == 0:
115                 cols.insert(0,'SWEEP')
116             #sweep variable repeats for each data block
117             #only need its first appearance
118             if hgi == 0 and li > 0:
119                 cols = cols[1:]
120             #concatenate columns from multiple lines
121             hline_single.extend(cols)
122         #save each complete header line, split column-wise
123         hlines.append(hline_single)
124     #regroup header lines into columns
125     hlines = list(zip(*hlines))
126     #return the flattened columns
127     return [','.join(col) for col in hlines]
128
129
130 def get_record_data(raw_rec, rec_profile):
131     """
132 ----Extract data from a record, convert to floats, and group by column
133
134 ----Params:
135 ----------
136 ----raw_rec: list returned by read_record
137 ----rec_profile: dict returned by get_rec_profile
138
139 ----Returns:
140 ----List of dicts, each contains data for a single MC instance.
141 ----Keys of each dict are:
142 --------mcind: int, MC instance index number
143 --------age: float, MOSRA age in s
144 --------sweep: tuple, sweep variable values
145 --------meas: list of tuples. Each tuple is a data column:
146 ----------------measured values for a variable for each sweep variable value
147 ----Empty list in case of malformed record.
148 ----"""
149     return_list = []
150
151     #check for a malformed record. Seems to happen at end of DC post-stress simulation
152     try:
153         malform = raw_rec[rec_profile['datastarts'][-1][-1]]
154     except IndexError:
155         return []
156
157     #get MOSRA age. Value is missing if record is from a pre-stress simulation.
158     #there is only one per record in all cases
159     try:
160         mosra_age = float(raw_rec[rec_profile['age'][0]].split()[-1])
161     except IndexError:
162         mosra_age = 0.0
163
164     #there are multiple subrecords, one for each MC instance if this was a post-stress simulation.
165     #there is only one if this was a combined pre and post stress simulation, or a pre-stress simulation.
```

```
166     mcline_inds = rec_profile['mcind']
167     #cover non Monte Carlo case
168     if not mcline_inds: mcline_inds = iter(range(-1,-2,-1))
169     for rec_ind,mc_ln in enumerate(mcline_inds):
170         subrec = {'mcind':[],'age':mosra_age,'sweep':[],'meas':[]}
171         if mc_ln < 0:
172             subrec['mcind'] = 1#non MC case
173         else:
174             subrec['mcind'] = int(raw_rec[mc_ln].split()[-2])
175         for strt,stp in zip(rec_profile['datastarts'][rec_ind],rec_profile['dataends'][rec_ind]):
176             cols = list(zip(*[map(engr_to_float,row.split()) for row in raw_rec[strt:stp]]))
177             subrec['meas'].extend(cols[1:])
178         subrec['sweep'] = cols[0]
179         return_list.append(subrec)
180     return return_list
181
182 def engr_to_float(num_str):
183     """
184 ----Convert a string representation of a number in engineering format to its float.
185
186 ----Params:
187 ----------
188 ----num_str: string. Number in engineering format
189
190 ----Returns:
191 -----------
192 ----float
193 ----"""
194     mag_char = num_str[-1]
195     if mag_char in 'fF':
196         mag = 'E-15'
197     elif mag_char in 'pP':
198         mag = 'E-12'
199     elif mag_char in 'nN':
200         mag = 'E-09'
201     elif mag_char in 'uU':
202         mag = 'E-06'
203     elif mag_char in 'mM':
204         mag = 'E-03'
205     elif mag_char in 'kK':
206         mag = 'E+03'
207     elif mag_char in 'xX':
208         mag = 'E+06'
209     elif mag_char in 'gG':
210         mag = 'E+09'
211     else:
212         mag = 'E+00'
213     return float(num_str[:-1]+mag)
214
215
216 ################################################
217 ################################################
218 #BEGIN SCRIPT
219
220 parser = argparse.ArgumentParser(description="""Extract Reliability Analysis results printed to a listing
        file.
```

```
221 ------------------------------------------------Verified to work with TRAN or DC analysis. DC analysis
        must be post-aging only.
222 ------------------------------------------------Works for analysis with and without Monte Carlo.
223
224 ------------------------------------------------Variables created:
225 ------------------------------------------------sweep: independent sweep variable vector
226 ------------------------------------------------age: MOSRA age vector
227 ------------------------------------------------meas: 4D array of measurement data
228 ------------------------------------------------D1: variable index
229 ------------------------------------------------D2: sweep index
230 ------------------------------------------------D3: age index
231 ------------------------------------------------D4: Monte Carlo instance index
232 ------------------------------------------------sweep_hdr: header string for indep
233 ------------------------------------------------meas_hdr: header strings for meas
234 ------------------------------------------------""")
235 parser.add_argument('lisfile',type=Path,help='Listing file containing analysis printouts.')
236 parser.add_argument('-o',type=Path,help='Output file name. Default is same as lisfile. Do not include an
        extension.')
237 parser.add_argument('-f',type=str,choices=['m','n','b'],default='b',help='Output format. Options are m for
        MATLAB, n for NumPy, b for both (default).')
238 args = parser.parse_args()
239 #set default output name if needed
240 if not args.o:
241     args.o = args.lisfile
242
243 age_list = np.array([])
244 append_age = True
245
246 with open(args.lisfile,'rt') as lisfile:
247     for rec_ind,rec_raw in enumerate(read_record(lisfile)):
248         if rec_ind == 0:
249             rec_profile = get_rec_profile(rec_raw)
250             headerstrings = get_headers(rec_raw, rec_profile)
251             #post-stress sim has different format than pre+post stres sim
252             #post-stress only: same format throughout, iterates over MC instance, then MOSRA age
253             #pre+post stress: iterates over age then MC instance. Inner loop begins with pre-stress format
254             #then switches to post-sress format. Pattern repeats on each MC instance.
255             #Age not recorded for pre-stress simulation
256             poststress_sim = bool(rec_profile['age'])
257         if not poststress_sim:
258             #Record age only during 1st MC index, it repeats after
259             #Re-profile the record if the format is expected to change
260             if append_age:
261                 #skip if non Monte Carlo
262                 if rec_profile['mcind']:
263                     if rec_ind == 0:
264                         prev_mcind = int(rec_raw[1].split()[-2])
265                     curr_mcind = int(rec_raw[1].split()[-2])
266                     append_age = (curr_mcind == prev_mcind)
267                     prev_mcind = curr_mcind
268                 #switched MC instance, record format change
269                 if not append_age:
270                     rec_profile = get_rec_profile(rec_raw)
271                 if rec_ind == 1:
272                     rec_profile = get_rec_profile(rec_raw)
273             else:
```

```
274                 if rec_ind%len(age_list) < 2:
275                     rec_profile = get_rec_profile(rec_raw)
276
277         #import data
278         rec_data = get_record_data(rec_raw, rec_profile)
279         #skip malformed records
280         if not rec_data: continue
281         if append_age:
282                 age_list = np.append(age_list,rec_data[0]['age'])
283         #post-stress format: each record is for an age step and contains subrecords for each MC index
284         #These can be collated and stacked in 4D
285         #pre+post stress contain no subrecords, these are stacked in 3D and reshaped into 4D at the end
286         if poststress_sim:
287             data_buff = np.array([d['meas'] for d in rec_data], ndmin=4)
288         else:
289             data_buff = np.array(rec_data[0]['meas'], ndmin=3)
290         if rec_ind == 0:
291             meas_data = data_buff
292             sweep_vals = np.array(rec_data[0]['sweep'])
293         else:
294             meas_data = np.concatenate((meas_data, data_buff), axis=0)
295
296 if poststress_sim:
297     meas_data = np.moveaxis(meas_data,[0,1],[2,3])
298 else:
299     meas_data = np.moveaxis(meas_data,0,-1)
300     meas_data = np.reshape(meas_data,[*meas_data.shape[0:2],len(age_list),-1],order='F')
301
302 #export data
303 if args.f in 'mb':
304     scipy.io.savemat(Path(args.o.stem+'.mat'),{'meas_data':meas_data,'sweep':sweep_vals,'age':age_list,'
          meas_hdr':headerstrings[1:],'sweep_hdr':headerstrings[0]},appendmat=True,do_compression=True)
305 if args.f in 'nb':
306     np.savez_compressed(args.o.stem,meas_data=meas_data,sweep=sweep_vals,age=age_list,meas_hdr=
          headerstrings[1:],sweep_hdr=headerstrings[0])
```

Appendix H

hspice_collate_MC_RAN.py

Collate multiple output files generated by hspice_RAN_pull.py (appendix G). To be used if a Monte Carlo simulation was run in multiple parts.

Invocation: **hspice_collate_MC_RAN.py** [**-h**] *infiles outfile* **-f m|n|b**

- -h: help message
- infiles: NumPy (.npz) files to be collated
- outfile: output file name
- -f: output file format, m: MATLAB, n: NumPy, b: both

```
#!/home/adimopou/anaconda3/bin/python
import argparse
from pathlib import Path
import numpy as np
import scipy.io

##############################################
##############################################
#BEGIN SCRIPT

parser = argparse.ArgumentParser(description="""Collate hspice_RAN_pull.py outputs. To be used if Monte Carlo analysis was carried out in multiple pieces and data is spread over multiple files. Data structures in all files must be equal, only the values in meas_data may change between files.""")
parser.add_argument('infiles', type=Path, nargs='+', help="npz files with data to be collated. e.g. data*.npz")
parser.add_argument('outfile',type=Path, help="output file name")
parser.add_argument('-f',type=str,choices=['m','n','b'],default='b',help='Output format. Options are m for MATLAB, n for NumPy, b for both (default).')
```

```
15 args = parser.parse_args()
16
17 for f_ind,f_name in enumerate(args.infiles):
18     with np.load(f_name) as Dbuff:
19         if f_ind == 0:
20             meas_data = Dbuff['meas_data']
21             if meas_data.ndim < 4:
22                 meas_data = np.reshape(meas_data, (*meas_data.shape,1))
23             sweep = Dbuff['sweep']
24             age = Dbuff['age']
25             meas_hdr = Dbuff['meas_hdr']
26             sweep_hdr = Dbuff['sweep_hdr']
27         else:
28             meas_data_2 = Dbuff['meas_data']
29             if meas_data_2.ndim < 4:
30                 meas_data_2 = reshape(meas_data_2, (*meas_data_2.shape,1))
31             meas_data = np.concatenate((meas_data,meas_data_2), axis=3)
32
33 #export data
34 if args.f in 'mb':
35     scipy.io.savemat(Path(args.outfile.stem+'.mat'),{'meas_data':meas_data,'sweep':sweep,'age':age,'
          meas_hdr':meas_hdr,'sweep_hdr':sweep_hdr},appendmat=True,do_compression=True)
36 if args.f in 'nb':
37     np.savez_compressed(args.outfile.stem,meas_data=meas_data,sweep=sweep,age=age,meas_hdr=meas_hdr,
          sweep_hdr=sweep_hdr)
```

Appendix I

tharcdc_waldtest.m

MATLAB script to perform the analysis described section 6.3 which leads to figure 6.7. This script will not run as is, the variable RMFILES on line 16 must first be defined. RMFILES is a directory listing of read_mode HSPICE simulation results which has been processed by hspice_collate_MC_RAN.py. Code for generating plots of the analysis results is not included.

```
1  %%%%%%%%%%%%%%%%%%
2  %%CONSTANTS
3  %%%%%%%%%%%%%%%%%%
4  HR = 3600;
5  DY = 24*HR;
6  YR = 365*DY;
7  MTH = YR/12;
8  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
9  %%LOAD READMODE SENSOR DATA
10 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
11 RMDATA = table;
12 RMDATA.T = zeros(0);
13 RMDATA.VDD = zeros(0);
14 RMDATA.IIN = {};
15 RMDATA.Properties.VariableUnits = ["C","V","A"];
16 %RMFILES: directory list of files generated by hspice_collate_MC_RAN.py
17 %IIN matrix: VINxAGExMC_INDEX
18 %VIN was swept 0:0.1:1.8
19 for fidx = 1:numel(RMFILES)
20     Dbuff = load(fullfile(RMFILES(fidx).folder,RMFILES(fidx).name));
21     if fidx == 1
22         Dbuff.meas_hdr = string(Dbuff.meas_hdr);
23         iin_idx = contains(Dbuff.meas_hdr,"current")&contains(Dbuff.meas_hdr,"vin");
24         numage = numel(Dbuff.age)-1;
25         age_rr = linspace(Dbuff.age(1),numage*diff(Dbuff.age(1:2)),numage);
26         vsweep_rr = Dbuff.sweep;
27     end
28     tv_set = regexp(RMFILES(fidx).name,'_T(?<temp>\d+)_VDD(?<vdd>\d+)','names');
```

```
29      RMDATA.T(fidx) = str2double(tv_set.temp);
30      RMDATA.VDD(fidx) = 1e-3*str2double(tv_set.vdd);
31  end
32  %Mark time offset to account for burnin testing
33  %TSMC burnin: 6 hours @ 1.4VDD core and 1.1VDD I/O, 25C
34  %Approx equivalent to 1 month at 1.8 V VDDIO and 25 C
35  burnin0_t_idx = interp1(age_rr,1:numel(age_rr),YR*1/12,'nearest');
36  burnin0_mu = RMDATA.IIN{RMDATA.T==25 & RMDATA.VDD==1.8}(end,burnin0_t_idx,1);
37  RMDATA.burnin_idx = cellfun(@(x) interp1(squeeze(x(end,:,1)),1:numel(age_rr),burnin0_mu,'nearest'),RMDATA.
        IIN);
38  RMDATA.burnin_mu0 = cellfun(@(x,y) x(end,y,1),RMDATA.IIN,num2cell(RMDATA.burnin_idx));
39  clear Dbuff tv_set numage;
40  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
41  %%PERFORM WALD TEST
42  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
43  WTRESULTS = table;
44  WTRESULTS.T = RMDATA.T;
45  WTRESULTS.VDD = RMDATA.VDD;
46  WTRESULTS.results = cell(height(WTRESULTS),0);
47
48  %zero measurements to burnin
49  %sweep sample size from 50:1000
50  %test at alpha = 0.1,0.05,0.01
51  for t_idx = 1:height(WTRESULTS)
52      b_idx = RMDATA.burnin_idx(t_idx);
53      idata = squeeze(RMDATA.IIN{t_idx}(end,b_idx:end,:));
54      %age 0 = time after burn-in
55      wtage = age_rr(b_idx:end) - age_rr(b_idx);
56      WTRESULTS.results{t_idx} = wtest(idata,wtage,burnin0_mu,50:10:1000,[0.1  0.05  0.01]);
57  end
58  %Now plot results
59  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
60  %%WALD TEST FUNCTIONS
61  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
62  function waldReject = wtest(Idata, age_vec, mu0, sampsizes, sig_alpha)
63      %perform Wald test.
64      %Determine minimum sample size for the test to correctly reject,
65      %and the minimum age at which this rejection occurs.
66      %
67      %Idata: current data samples, matrix of size NAGE x NMonteCarlo
68      %age_vec: ages at which indata was recorded, vector length NAGES
69      %mu0: true mean at t=0
70      %sampsizes: number of indata samples to use starting from index 1, scalar
71      %or vector of length NSAMP
72      %sig_alpha: significance level for sample mean CI, scalar or vector of
73      %length NALPHA
74      %
75      %waldReject: Minumum sample sizes required for Wald test to reject H0
76      %and ages at which these rejections occur. cell
77
78      sampsizes = sort(sampsizes,'descend');
79
80      NAGES = numel(age_vec);
81      NSAMPSIZES = numel(sampsizes);
82      NALPHA = numel(sig_alpha);
83
```

```
84      %for each age and alpha, record min sample size for test to reject H0
85      %NaN = fails to reject at largest sample size
86      waldResult = nan(NAGES,NALPHA);
87
88      stoptesting_alpha = [];
89      sidx = 1;
90      for tidx = 1:NAGES
91          %perform Wald test, vary sample size from largest to smallest
92          %stop when the test fails to reject H0. Previous sample size is min
93          %required for test to reject H0 at this age.
94          %all alpha values will not necessarily be tested for each sample
95          %size since the Wald test will reject at different sample sizes for
96          %different alpha levels.
97
98          %stop testing at an alpha level if the test has used the min sample size
99          %If the test rejected, it will also reject at all later ages
100         %If the test accepted, it will begin rejecting at later ages
101         alpha_to_test = setdiff(sig_alpha,stoptesting_alpha);
102         if isempty(alpha_to_test)
103             break;
104         end
105         %alpha_to_test is a subset of sig_alpha, need a map between the two
106         alpha_idx_map = arrayfun(@(x) find(sig_alpha == x),alpha_to_test);
107
108         while(sidx <= NSAMPSIZES)
109             %confidence interval
110             [UC,~,~,~] = bootstrap_CI(Idata(tidx,:),sampsizes(sidx),alpha_to_test);
111             %wald test
112             %UC.buc: vector with bootstrap upper conf bound for each
113             %alpha_to_test
114             wt_alpha_accept_idx = mu0 < UC.buc;
115
116             %largest sample gives narrowest CI, decreasing sample size
117             %(increasing CI) will not lead to a rejection so stop testing
118             if sidx == 1
119                 if nnz(wt_alpha_accept_idx) == numel(alpha_to_test)
120                     break;
121                 else
122                     alpha_to_test = alpha_to_test(~wt_alpha_accept_idx);
123                     alpha_idx_map = arrayfun(@(x) find(sig_alpha == x),alpha_to_test);
124                 end
125             %reached the smallest sample size, if the test has rejected all
126             %the way to this point (widest CI), it will reject for all
127             %later ages, there is no point in testing a later ages
128             elseif sidx == NSAMPSIZES
129                 wR_idx = alpha_idx_map(~wt_alpha_accept_idx);
130                 waldResult(tidx,wR_idx) = sampsizes(sidx);
131                 wR_idx = alpha_idx_map(wt_alpha_accept_idx);
132                 waldResult(tidx,wR_idx) = sampsizes(sidx-1);
133                 stoptesting_alpha = [stoptesting_alpha,alpha_to_test];
134             %Normal case. find the sample size where the test switches from
135             %reject to accept. This is the min sample needed to id this
136             %age. Stop testing here for the given alpha level.
137             else
138                 wR_idx = alpha_idx_map(wt_alpha_accept_idx);
139                 waldResult(tidx,wR_idx) = sampsizes(sidx-1);
```

```
140                 alpha_to_test = alpha_to_test(~wt_alpha_accept_idx);
141                 if numel(alpha_to_test) == 0
142                     break;
143                 end
144                 alpha_idx_map = arrayfun(@(x) find(sig_alpha == x),alpha_to_test);
145             end%if/else sidx size
146             sidx = sidx+1;
147         end%sidx loop
148         %Expect age vs sample size to be monotonic so try to save some effort by not exhastively testing
                all sample
149         %sizes at each age. Keep decreasing the initial sample size as age
150         %increases.
151         sidx = min(waldResult(tidx,:));
152         if isnan(sidx)
153             sidx = 1;
154         end
155     end%tidx loop
156     %remove duplicate entries in each column of waldResult, keep only the
157     %earlieast occurence of any given sample size.
158     for col_idx = 1:NALPHA
159         wRcol = waldResult(:,col_idx);
160         uvcol = unique(wRcol);
161         uvcol = uvcol(~isnan(uvcol));
162         keep_idx = arrayfun(@(x) find(wRcol==x,1),uvcol);
163         remove_idx = setdiff(1:numel(wRcol),keep_idx);
164         waldResult(remove_idx,col_idx) = nan;
165     end
166     %only return non-nan values
167     return_idx = ~isnan(waldResult);
168     age_vec = age_vec(:);
169     return_ages = cellfun(@(x) age_vec(x),num2cell(return_idx,1),'UniformOutput',false);
170     return_sampsizes = cellfun(@(x,y) x(y),num2cell(waldResult,1),num2cell(return_idx,1),'UniformOutput',
            false);
171     waldReject = struct('age',return_ages,'sampsize',return_sampsizes);
172 end%function wtest
173
174
175 function [UC,LC,x_m,x_s] = bootstrap_CI(indata,sampsize, sig_alpha)
176     %compute the confidence inteval on the sample mean as well as bootstrap
177     %estimates of the confidence interval on the upper and lower bounds of
178     %the sample mean CI
179     %indata: data samples, matrix of size NAGE x NMonteCarlo
180     %sampsize: number of indata samples to use starting from index 1, scalar
181     %or vector of length NSAMP
182     %sig_alpha: significance level for sample mean CI, scalar or vector of
183     %length NALPHA
184     %NOTE: bootstrap CI computed at alpha=0.01
185     %
186     %UC: upper confidence bound, struct with fields:
187     %   xuc: sample mean upper conf bound, matrix size NAGE x NALPHA x
188     %   NSAMP
189     %   buc: bootstrap estimate of upper bound of 0.01 CI on xuc, matrix of
190     %   size NAGE x NALPHA x NSAMP
191     %   blc: boostrap estimate of lower bound of 0.01 CI on xuc, matrix of size NAGE x NALPHA x NSAMP
192     %LC: lowr confidence bound, struct with fields:
193     %   xlc: sample mean lower conf bound, matrix size NAGE x NALPHA x
```

```
194       %   NSAMP
195       %   buc: bootstrap estimate of upper bound of 0.01 CI on xlc, matrix of
196       %   size NAGE x NALPHA x NSAMP
197       %   blc: boostrap estimate of lower bound of 0.01 CI on xlc, matrix of size NAGE x NALPHA x NSAMP
198       %x_m: sample mean, matrix of size NAGE x NSAMP
199       %x_s: sample standard deviation, matrix of size NAGE x NSAMP
200
201       [NAGE,NMC] = size(indata);
202       NSAMP = numel(sampsize);
203       NALPHA = numel(sig_alpha);
204
205       %Bootstrap
206       NBOOT = 1000;%bootstrap resample size
207       BOOTALPHA = 0.01;%bootstrap CI significance level
208
209       %preallocate memory
210       x_m = nan(NAGE,NSAMP);
211       x_s = nan(NAGE,NSAMP);
212       UC.xuc = nan(NAGE,NALPHA,NSAMP);
213       UC.buc = nan(NAGE,NALPHA,NSAMP);
214       UC.bluc = nan(NAGE,NALPHA,NSAMP);
215       LC.xlc = nan(NAGE,NALPHA,NSAMP);
216       LC.buc = nan(NAGE,NALPHA,NSAMP);
217       LC.blc = nan(NAGE,NALPHA,NSAMP);
218
219       for sidx = 1:NSAMP
220           %confidence interval on sample mean
221           n_samp = sampsize(sidx);
222           x_m(:,sidx) = mean(indata(:,1:n_samp),2);
223           x_s(:,sidx) = std(indata(:,1:n_samp),0,2);
224           [LC.xlc(:,:,sidx),UC.xuc(:,:,sidx)] = normconf_interval(x_m(:,sidx),x_s(:,sidx),n_samp,sig_alpha);
225           %bootstrap confidence interval on sample mean CI
226           xstar_LC = nan(NAGE,NALPHA,NBOOT);
227           xstar_UC = nan(NAGE,NALPHA,NBOOT);
228           for bidx = 1:NBOOT
229               %bootstrap replication, independent for each age
230               xstar = cell2mat(cellfun(@(z) z(randi(n_samp,1,n_samp)),num2cell(indata(:,1:n_samp),2),'
                      UniformOutput',false));
231               xstar_m = mean(xstar,2);
232               xstar_s = std(xstar,0,2);
233               [xstar_LC(:,:,bidx),xstar_UC(:,:,bidx)] = normconf_interval(xstar_m,xstar_s,n_samp,sig_alpha);
234           end%bidx
235           %avoid loops over NALPHA. split xstar_LC and xstar_UC into cell arrays
236           xstar_LC_C = mat2cell(xstar_LC,ones(1,NAGE),ones(1,NALPHA),NBOOT);
237           xstar_UC_C = mat2cell(xstar_UC,ones(1,NAGE),ones(1,NALPHA),NBOOT);
238           %also need to split LC and UC
239           xlc_C = num2cell(squeeze(LC.xlc(:,:,sidx)));
240           xuc_C = num2cell(squeeze(UC.xuc(:,:,sidx)));
241           %use pivot confidence interval since bootstrap resample has unknown
242           %(but should be approx normal) distribution
243           [LC.blc(:,:,sidx),LC.buc(:,:,sidx)] = cellfun(@(z,zstar) pivotconf_interval(z,squeeze(zstar),
                  BOOTALPHA),xlc_C,xstar_LC_C);
244           [UC.blc(:,:,sidx),UC.buc(:,:,sidx)] = cellfun(@(z,zstar) pivotconf_interval(z,squeeze(zstar),
                  BOOTALPHA),xuc_C,xstar_UC_C);
245       end%sidx
246   end
```

```
247
248 function [LC,UC] = normconf_interval(x_mu,x_s,n_samp, sig_alpha)
249     %return lower and upp limits of normal confidence interval
250     %x_mu: sample mean
251     %x_s: sample standard deviation
252     %n_samp: sample size
253     %sig_alpha: significance level
254     %confidence interval covers 1-sig_alpha:
255     % 0.01 alpha -> 99% confidence interval
256     LC = x_mu - norminv(1 - sig_alpha/2).*x_s./sqrt(n_samp);
257     UC = x_mu + norminv(1 - sig_alpha/2).*x_s./sqrt(n_samp);
258 end
259
260 function [LC,UC] = pivotconf_interval(T,Tstar,sig_alpha)
261     %return lower and upper limits of pivotal bootstrap confidence interval
262     %T: statistic, scalar
263     %Tstar: bootstrap replications of T, must be a vector
264     %confidence interval covers 1-sig_alpha:
265     % 0.01 alpha -> 99% confidence interval
266     [F,x] = ecdf(Tstar);
267     lower_alpha_i = interp1(F,1:numel(F),1-sig_alpha/2,"nearest");
268     upper_alpha_i = interp1(F,1:numel(F),sig_alpha/2,"nearest");
269     LC = 2*T - x(lower_alpha_i);
270     UC = 2*T - x(upper_alpha_i);
271 end
```