# Automatic Tuning of Digital Circuits

by

**Nurrachman Chih Yeh Liu**

**A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering)
in The University of Michigan
2011**

**Doctoral Committee:**

**Professor David Blaauw, Chair**

**Professor Dennis M. Sylvester**

**Assistant Professor James W. Cutler**

**Assistant Professor Zhengya Zheng**

**© Nurrachman Chih Yeh Liu 2011**

*To my family:*

*mom, your tireless concern and care*

*dad, your invaluable listening and support*

*bro, your patient discussions and advice*

# ACKNOWLEDGEMENTS

I have learned much from my graduate school career. My graduate school career has been anchored and pivotally advised by Professors David Blaauw and Dennis Sylvester. My advisor David has very patiently guided me throughout my research and generously shared his experience and knowledge. He has offered the key ideas that form the foundation for each and every step of the work. Without his patient and steadfast support, I could not have accomplished what I have today. In addition, Dennis has taught me tremendously, and shared his invaluable expertise and advice. Dennis has given vital advice and insight on every aspect, from projects to papers to ideas. I couldn't have completed my graduate school studies without their unwavering help and support.

I would like to thank Professor Zhengya Zheng and Professor James Cutler for graciously agreeing to be on my defense committee. I appreciate their guidance and time, and their enthusiastic involvement. During my graduate career, I've also had the fortunate opportunity to spend eight months interning at Qualcomm Corporate R&D, designing new variation detecting circuits for specific on-chip variation issues. I learned a great deal and would like to thank my manager Kendrick Yuen for his warm mentorship and involvement. Additionally, I would like to thank Dennis, and Shekhar Borkar of Intel for kindly allowing me to use their figures in the first chapter.

On my path towards graduate school, I've been fortunate to have been set towards the right research direction and experience by having the opportunity to do undergraduate research at the University of California, Berkeley, with Professor Jan Rabaey at the Berkeley Wireless Research Center and Professor Andrew Neureuther. I sincerely thank them both for their kind guidance and assistance. I also worked on undergraduate research with Lecturer Dan Garcia, whom I thank for his mentorship. Additionally, I'd like to thank Yuen Hui Chee for his mentorship during my undergraduate research.

I would also like to thank my many colleagues with whom I have collaborated so much. Scott Hanson for his mentorship and initial ideas on the transition detection circuit, and his valuable suggestions that helped form OxiGen; Prashant Singh for always lending a helping hand with problems, giving great suggestions, and for discussing ideas; Sudhir Satpathy for his cheerful enthusiasm in discussing research ideas, helping with issues, and bouncing problems off of; Dave Fick for his help in developing the Razor core, into which the Time-to-Digital Converter was embedded; Nathaniel Pinckney for his kind assistance with OxiGen; and David Lin and Li Li for their advice on analog circuits.

I also owe so much to my family: my parents who have selflessly always put me first, listened patiently, and given me so much; my mom Mary who has never ceased to support me and give me strength; my dad Charles who has always listened to my ramblings and shared his wisdom; and my brother, Christianto Liu, who has always taken the time to share his many experiences and given me sage advice for just about everything.

# TABLE OF CONTENTS

DEDICATION ........ ii

ACKNOWLEDGEMENTS ........ iii

List of Figures ........ vii

List of Tables ........ x

ABSTRACT ........ xi

CHAPTER I. Introduction ........ 1

1.1 Impact of process variation on circuit manufacture ........ 2

1.2 Implications of process variation on digital circuit design ........ 6

1.3 Main contribution of this work and organization of this work ........ 11

CHAPTER II. A 5 ps Time to Digital Converter for Measuring Timing in a Razor-type System ........ 14

2.1 Introduction to time to digital conversion ........ 15

2.2 Concepts of proposed time to digital converter ........ 18

2.3 Circuit implementation of time to digital converter ........ 21

2.4 Circuit techniques used for improving time to digital converter accuracy ........ 24

2.5 TDC calibration scheme ........ 25

2.6 Silicon implementation and measurement results ........ 26

2.7 Summary and discussion ........ 30

CHAPTER III. OxID: On-chip One-time Random ID Generation using Oxide Breakdown ....32
3.1 Introduction to oxide breakdown .... 33
3.2 Introduction to on-chip chip-ID systems.... 34
3.3 On-Chip vs Off-Chip ID Generation.... 36
3.4 Concepts of proposed OxID on-chip chip-ID system .... 37
3.5 Implementation of OxID circuit and system .... 41
3.6 Measurement results and discussion .... 42
3.7 Summary and discussion.... 51
CHAPTER IV. OxiGen: A True Random Number Generator using Time-Dependent Dielectric Breakdown ....52
4.1 Introduction to Random Number Generation.... 53
4.2 Hardware True Random Number Generation .... 56
4.3 OxiGen Operation .... 60
4.4 Measured Results .... 65
4.5 Conclusion.... 75
CHAPTER V. Mitigating and Monitoring Variation in Subthreshold ....76
5.1 Proposed Methods of Increasing Robustness to Variability in Subthreshold Circuit Design.... 77
5.2 Application of PVT Mitigation Design Techniques: 100% Self-Timed Transition Detection Circuit .... 80
CHAPTER VI. Conclusion....90
REFERENCES ....96

# List of Figures

Figure 1.1: Impact of channel length variation on frequency and power. ...........................3

Figure 1.2: Impact of channel length on frequency and power with leakage included. ......4

Figure 1.3: 6-T SRAM Cell .........................................................................................5

Figure 1.4: Illustration of Low-Vt usage being more power-efficient than sizing up regular-Vth transistors. .........................................................................8

Figure 1.5: Measured Results of FBB vs NBB vs ZBB chips. ...........................................9

Figure 1.6: Leakage Current vs Reverse Body Bias Voltage. ............................................9

Figure 1.7: Illustration of how FBB and RBB can tighten up the frequency distribution. 10

Figure 1.8: Illustration of NBB yield vs ABB yield vs WID-ABB yield. .........................11

Figure 2.1: Basic Start-Delayed TDC ..................................................................15

Figure 2.2: Basic Data-Delayed TDC ...................................................................16

Figure 2.3: Vernier start-delayed TDC (top); Vernier data-delayed TDC (bottom)..........17

Figure 2.4: TDC Die photo in addition to test core and configuration core. .....................19

Figure 2.5: Reference counter noise vs count value (top); reference count vs trial number. ......................................................................................20

Figure 2.7: TDC layout without high-metal power straps and decap (omitted for clarity). ......................................................................................22

Figure 2.8: Circuit implementations of key components of the TDC block......................24
Figure 2.9: Recorded bit counts for each TDC element (top); corresponding data and clock caps used for each bit...............................................................28
Figure 2.10: Output codes corresponding to input delay using reference delay chain (top);Histogram of output codes showing low noise.....................................29
Figure 2.11: Slack of Alpha core critical paths measured by the TDC. ...........................30
Figure 3.1: System architecture with individual array layout..........................................38
Figure 3.2: Schematic architecture of a single array.......................................................39
Figure 3.3: Schematic architecture of a single array with schematics of key components...........................................................................................41
Figure 3.4: Physical layout for a single core.....................................................................42
Figure 3.5: Hamming distance results for global algorithm. ............................................43
Figure 3.6: Hamming distance results for canary algorithm............................................43
Figure 3.7: Number of self-bit flips (robustness) vs $V_{DD}$ variation for 14 arrays.............44
Figure 3.8: Sense amplifier read margin (robustness) over Temperature for 14 arrays. ...45
Figure 3.9: Spatial distribution of bit probabilities for global algorithm (top) and canary algorithm (bottom)..................................................................46
Figure 3.10: Spatial distribution of the breakdown time for a typical oxide. ...................47
Figure 3.11: Stress intervals required for global (top) and canary (bottom) algorithms. .49
Figure 3.12: Die photo. .....................................................................................................50
Figure 4.1: Conceptual system diagram...........................................................................62
Figure 4.2: Circuit diagram of cell array, showing 3-T cells, sense amps, and drivers....62
Figure 4.3: NIST pass rates as a function of bits truncated. ............................................66

Figure 4.4: Stream pass rates for NIST-required large sequence lengths. ........................66
Figure 4.5: Spectral tests [49] of a pseudorandom binary sequence generated from a 7-bit LFSR, digits of π, and values from OxiGen before and after truncation. .67
Figure 4.6: Matrix of 300 kb each for pseudorandom binary sequence generated from a 7-bit LFSR, digits of π, and values from OxiGen before and after truncation. Bits were placed consecutively from top to bottom then left to right. ..........................................................................................................68
Figure 4.7: Sa_vref and vddh for single cell during generation of 500Mb data. A typical iteration yields 20 bits. ......................................................................69
Figure 4.8: Accelerated stress testing diagrams. ............................................................70
Figure 4.9: This cell exhibits constant bit production and the algorithm does not need to adjust stress voltage. ...........................................................................71
Figure 4.10: Die photo of OxiGen. ..........................................................................73
Figure 5.1: A traditional implementation of a high-input CMOS transition detector with OR-tree ..................................................................................81
Figure 5.2: Sub-Vth optimized self-timed wide-or structure for or'ing transition inputs. 82
Figure 5.3: Linear increase in TD output pulse-width due to second incoming transition 84
Figure 5.4: TD Ouput Pulsewidth for later arrival times of 2nd Input Transition.............86
Figure 5.5: Self-timed Transition Detection stage with asynchronous feedback input from wide-or stage. ...........................................................................88
Figure 5.6: Entire 100% Self-timed Transition Detector 100% Robust for Sub-Vth. ......89

# List of Tables

Table 3.1: Comparison with previous works. ....................................................................48

Table 3.2: Chip summary. ............................................................................................50

Table 4.1: Table of all NIST 800-22 tests, sequence lengths, sequence sample size (number of sequences), proportions of sequences passed, and minimum passing rate, using random bits generated by OxiGen after 4-bit truncation...72

Table 4.2: Summary of OxiGen technical specifications. ................................................73

Table 4.3: Table comparing OxiGen to prior true random number generators. ...............74

# ABSTRACT

Variation in transistors is increasing as process technology transistor dimensions shrink. Compounded with lowering supply voltage, this increased variation presents new challenges for the circuit designer. However, this variation also brings many new opportunities for the circuit designer to leverage as well.

We present a time-to-digital converter embedded inside a 64-bit processor core, for direct monitoring of on-chip critical paths. This path monitoring allows the processor to monitor process variation and run-time variations. By adjusting to both static and dynamic operating conditions the impact of variations can be reduced. The time-to-digital converter achieves high-resolution measurement in the picosecond range, due to self-calibration via a self-feedback mode. This system is implemented in 45nm silicon and measured silicon results are shown. We also examine techniques for enhanced variation-tolerance in subthreshold digital circuits, applying these to a high fan-in, self-timed transition detection circuit that, due to its self-timing, is able to fully compensate for the large variation in subthreshold.

In addition to mitigating variations we also leverage them for random number generation. We demonstrate that the randomness inherent in the oxide breakdown process can be extracted and applied for the specific applications of on-chip ID generation and on-chip true random number generation. By using dynamic automated self-calibrating algorithms that tune and control the on-chip circuitry, we are able to achieve extremely

high-quality results. The two systems are implemented in 65 nm silicon. Measured results for the on-chip ID system, called OxID, show a high-degree of randomness and read-stability in the generated IDs, both primary prerequisites of a high-quality on-chip ID system. Measured results for the true random number generator, called OxiGen, show an exceptionally high degree of randomness, passing all fifteen NIST 800-22 tests for randomness with statistical significance and without the aid of a post-processor.

# CHAPTER I

# Introduction

Variations in process, supply voltage, and temperature (PVT) play an increasingly critical role in digital, analog, and mixed signal circuit design. Corner design has traditionally been used to take PVT effects into account, but with continued process scaling to 65 nm and smaller nodes, taking PVT effects into account has shifted from an important performance-ensuring optimization to a critical make-or-break role on the fabricated circuit's functionality and operation [1].

PVT exerts a wide-sweeping range of effects on digital circuit design, manufacture, operation, and performance. In this proposal, we first look at the challenges PVT brings to the design space, and then explore particular solutions to mitigate PVT variations during both the circuit design phase and in actual circuit operation. A specific focus is placed upon timing detection circuits used to detect and mitigate PVT variations, such as transition detection circuits and time to digital converters. In addition, we consider ways to exploit PVT for particular applications, during both the circuit design phase and during actual circuit use. Specifically, we examine Chip ID and random number generation, in our implementations called OxID and OxiGen, respectively, and

then broadly examine future ways in which PVT can be further mitigated and detected, with a design analysis example of a subthreshold transition detection circuit implemented in silicon.

## 1.1 Impact of process variation on circuit manufacture

Process variation effects during manufacturing impact the parametric and functional yield of chips. Parametric yield is the percentage of die that meet the required frequency and power specifications, while functional yield is simply the percentage of die that meet basic functionality. As technology scales to smaller nodes, these PVT effects that directly impact the parametric and functional yield become more significant. Their impact on the circuit design phase is ever-growing as designers now must account heavily for these effects through either margining, or system, architectural and circuit level techniques.

We first look at how process variation, in conjunction with voltage supply and temperature variation, impacts the parametric yield of dies, and why this impact is becoming more significant as technology scales. Amongst the plethora of parameters that vary due to process, the main parameters that strongly affect both performance and power (and thus the parametric yield) are channel length and threshold voltage (see Figure 1.1). With growing channel length and threshold voltage variation, logic blocks that fail timing will slow down the overall chip's clock frequency, and if any one logic block becomes too slow due to PVT variation, it may cause the reliable operating frequency of that chip

to fall below specification; that is, it will cause a parametric yield loss due to insufficient performance.

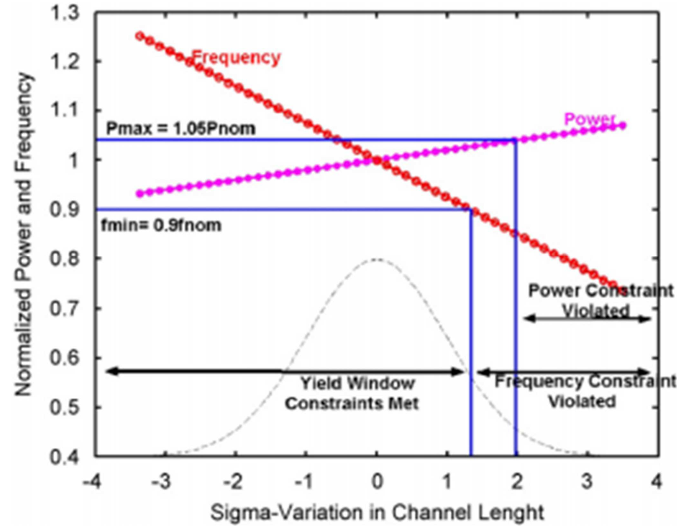



**Figure 1.1: Impact of channel length variation on frequency and power.**

In addition, as technology scales to ever smaller nodes, leakage power is becoming a significant if not dominant part of the power budget. The growing variation in channel length and threshold voltage translates to larger variation in the total leakage in a chip, due to the subthreshold current (off current) of the transistor being exponentially dependent on the threshold voltage and channel length. Combined with increasing gate leakage variation due to increasingly thinner gate oxides, these variations in leakage current (subthreshold and gate leakage) cause certain die to fail power budgets, therefore decreasing the parametric yield. Figure 1.2 from [2] illustrates this additional constraint imposed by leakage power on the power budget, as compared to Figure 1.1.

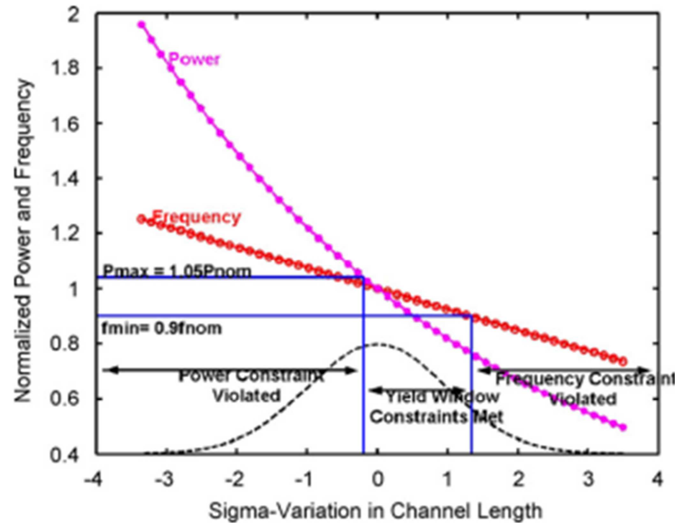



**Figure 1.2: Impact of channel length on frequency and power with leakage included.**

The second major impact on chip manufacturing due to process variation is functional yield. Circuits that depend strongly on transistor matching, such as ratioed-logic and analog circuits are strongly affected by transistor mismatch caused by process variation. While this transistor mismatch is due to many parameters, such as width and length mismatch, the most significant parameter is threshold voltage mismatch. As shown in [2], a good way of examining the growing impact of these random variations on functional yield is by examining SRAM yield in the newer technology nodes; the SRAM cell is precisely tuned for maximum yield due to its economic importance and consequence, and is perhaps the most fundamental circuit under which the effects of process variation and thus threshold voltage mismatch can be analyzed.

The 6-T SRAM cell consists of two back-to-back inverters, with their intermediate nodes accessed by 2 NMOS pass-gate transistors; see Figure 1.3. The delicate balancing act of transistor strength tuning required in SRAM functional yield

optimization falls in making sure that the read and write operations are functional under all PVT conditions and variation, an already difficult task further exacerbated by the opposing sizing constraints imposed by read and write. In addition, random dopant fluctuation (RDF) causes transistor threshold voltage variation, and is inversely proportional to the gate area. The high density requirement of SRAM cells results in extremely small gates and hence a high amount of RDF. These opposing constraints and pronounced RDF-induced threshold mismatches make designing the SRAM cell a highly optimized process, one that involves extensive width and length tuning, as these are the most effective knobs for tuning transistor strength because they indirectly tune the threshold voltage.

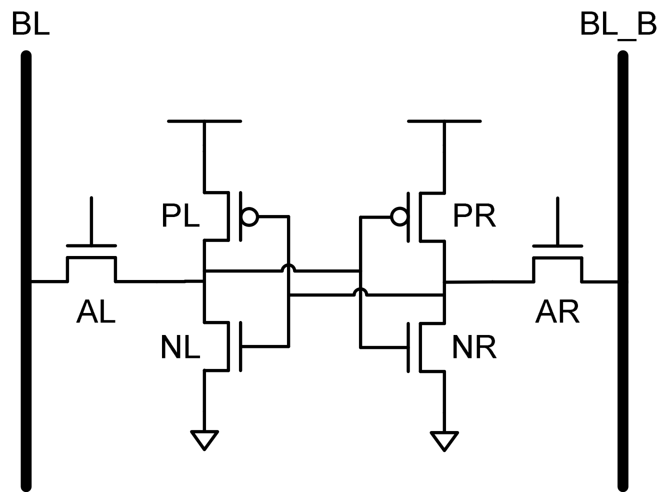



**Figure 1.3: 6-T SRAM Cell**

We now examine why the SRAM cell's functional yield so strongly depends on threshold mismatch. During a read operation, both bitlines are precharged, and one of them will discharge through the resistive division formed by the access transistor on that bitline and the NMOS pull-down of one of the inverters. If the NMOS pull-down is not

sufficiently strong, the resistive division formed between it and the access transistor may actually flip the cell contents during a read. This error may occur if random mismatch causes the threshold voltages to be skewed to favor such a scenario. During a write, the access transistor is required to be stronger in order to properly pull the internal node down to 0 during a 0-value write; there is a resistive division formed between the PMOS of the inverter and the access transistor. The access transistor must pull the internal node high enough that it triggers the flip-point of the inverter that the node is connected to. If threshold voltages happen to be skewed in such a way that the access transistor is not strong enough, then a write failure would occur. Thus, the opposite constraints imposed by read and write require the access transistor to be precisely balanced between being stronger than the PMOS, yet weaker than the NMOS. In the presence of millions of SRAM cells, where yield is required to be extremely high (such as 5-6 sigmas), any amount of significant PVT variation that is not properly accounted for during design can cause functional yield to be missed.

Therefore, because the SRAM cell requires extremely high density and thus has stringent constraints, it illustrates that in order for all complex circuits, including flip flops and latches, to achieve functional yield, careful simulation and analysis must be performed that takes into account all possible scenarios introduced by PVT effects.

## 1.2 Implications of process variation on digital circuit design

Due to the mentioned PVT factors that cause increasingly degrading yield losses, the circuit design phase must take PVT variation into account more strongly than before.

Traditional methods such as margining, at all levels (ie increasing sizing at the transistor level, relaxing timing constraints at the gate level, or relaxing system constraints at the system level), are becoming increasingly insufficient and inefficient at addressing the problems posed by PVT variations. More sophisticated approaches of addressing PVT and its variation must be used in conjunction with traditional methods; with today's high logic density, using some extra transistors for logic to directly monitor and conform to PVT and its variation may not only ease meeting yield but actually improve it. Improved performance, decreased power, and decreased failure rates are all achievable with some extra effort during the circuit design phase by adding extra circuitry or making more intelligent design choices. This section surveys a sampling of the ways in which such more intelligent design choices and smart circuitry can be implemented.

We first briefly overview two existing techniques that have been studied and analyzed in depth: multi-Vth or dual-Vth type methods; and body-biasing methods. Both of these methods have been analyzed extensively in the literature; ranging from dual-Vth energy comparisons [3,4] to body-biasing versus non-body biasing energy comparisons [3], to applications of body biasing for ultra low power chips [5].

Multi-Vth or its smaller subset, dual-Vth, uses a circuit optimization algorithm that exploits flexibility of today's modern processes. In its most simplified view, this approach simply seeks to cut down excess leakage power by assigning slower, non-critical gates a higher Vth, and also by lowering the Vth in the most critical paths to ensure that they do not need to be upsized by timing and place and route tools. The following Figure 1.4 taken from [3] shows that setting critical paths to *lower* Vth actually

lowers total power, simply because these critical path gates do not have to be upsized as much (upsizing translates to increased $CV_{DD}^2$ dynamic power).

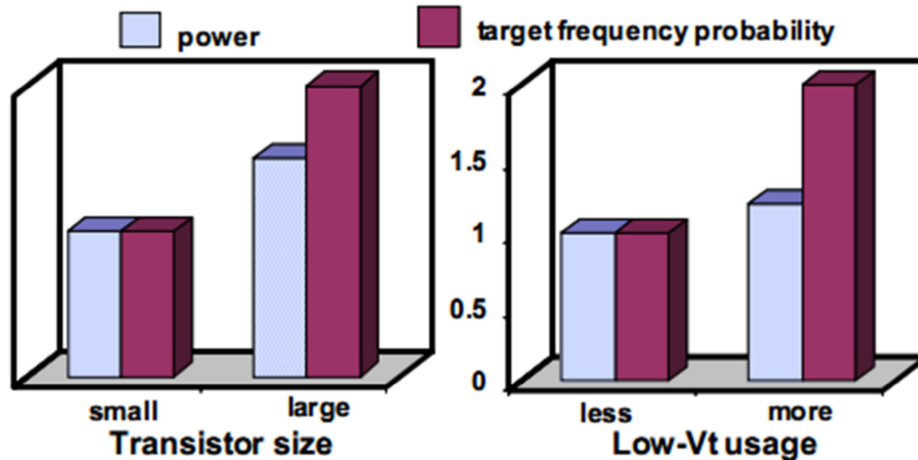



**Figure 1.4: Illustration of Low-Vt usage being more power-efficient than sizing up regular-Vth transistors.**

Body-biasing has also been extensively studied and analyzed, as it can offer performance boosts in conjunction with power reduction, simply by modulating the body biasing of groups of transistors. Intelligent approaches to grouping transistors into appropriate islands for body-biasing have been studied [6], showing that such an approach can indeed save power even with a few groups only, which is the most feasible in actual circuit implementations. Circuit implementations of entire ASICs have been implemented with and without body bias; Figure 1.5 taken from [3] compares forward body biasing an entire chip (FBB) versus its non-body biased counterpart (NBB) and even its zero-biased state (ZBB). The FBB chip can achieve much higher frequencies, especially at lower supply voltages where Vth modulation has more impact. Forward body biasing reduces the P-N depletion width, therefore making it easier for gate

depletion to occur (*decreasing* $V_{th}$) and decreasing short-channel effects (and thus *reducing Vth variation*).

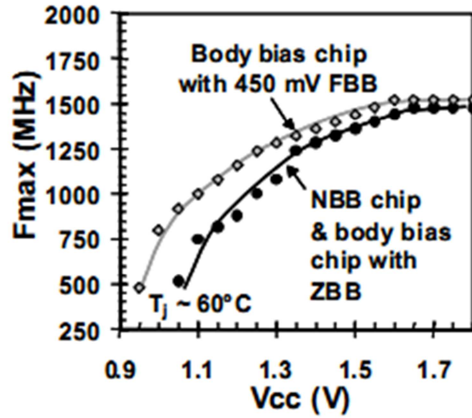



**Figure 1.5: Measured Results of FBB vs NBB vs ZBB chips.**

Simultaneously, reverse body biasing a chip can reduce standby leakage power dramatically, for the opposite reason as forward body biasing; reverse body biasing increases Vth. Figure 1.6, taken from [3] shows the measured reverse body-biased chip leakage current plotted with the worst-case-length simulated leakage current and the nominal-case-length simulated leakage current.

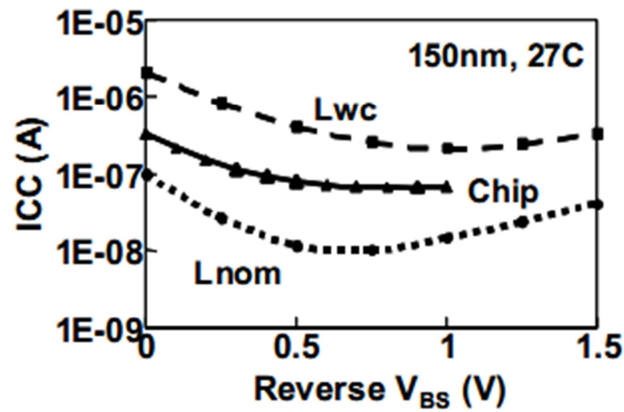



**Figure 1.6: Leakage Current vs Reverse Body Bias Voltage.**

Finally, as mentioned above, by combining forward body biasing (improved performance) and reverse body biasing (decreased leakage power) into adaptive body biasing, the benefits of both extremes can be obtained. There are two extremes to using adaptive body biasing: “static” or post-fabrication one-time calibrated body biasing, where slow parts are sped up with FBB, and fast parts are slowed down to reduce leakage, using RBB; and “dynamic”, or run-time calibrated body biasing, which requires the use of additional transistor circuitry and also design-phase intelligent algorithmic planning, both of which are necessary sacrifices as mentioned before. From Figure 1.7, taken from [3], even the simplistic “static” adaptive body biasing extreme can reduce die-to-die variation between chips dramatically, thus reducing binning spread and improving parametric yield.

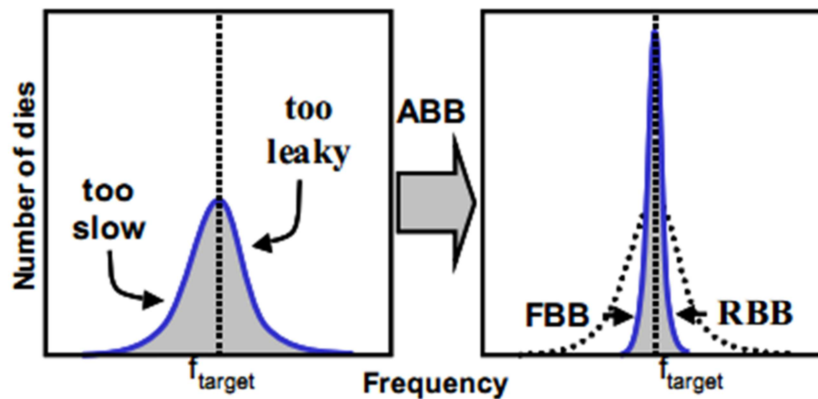



**Figure 1.7: Illustration of how FBB and RBB can tighten up the frequency distribution.**

An improvement on this simplistic “static” adaptive body-biasing is the two-way “bidirectional” static adaptive body biasing [3], where both PMOS and NMOS wells are biased separately. This bidirectional static ABB has been measured in [3] to reduce die-

to-die frequency variations (u/sigma) by an order of magnitude, and improving yield from 40% yield (no body bias) to 100%. Taken one step further, "WID-ABB" from [3] separately biases different groups or wells within the chip, to account for within-die variation (although still not at the extreme ABB end of "dynamic" run-time ABB). WID-ABB enhances on bidirectional ABB by moving all dies one bin up, to "bin 2" (as seen in Figure 1.8).

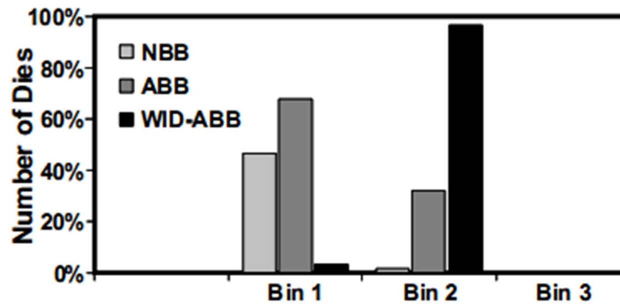



**Figure 1.8: Illustration of NBB yield vs ABB yield vs WID-ABB yield.**

## 1.3 Main contribution of this work and organization of this work

In this chapter, we discussed the basic effects of process, supply and temperature (PVT) variation and their corresponding impact on circuit manufacture (parametric and functional yield). We have also described how this impact on circuit manufacture has necessitated new approaches to increase yield and enhancing robustness, through means such as body biasing and multi-Vth approaches.

In this work, we propose novel ways of mitigating variation introduced by PVT, and also for exploiting this variation for application and circuit design purposes. These goals are accomplished through circuits specifically designed for automated tuning and calibration. We introduce an all-digital time-to-digital converter (TDC) that is accurate to the order of picoseconds; its novelty is using a statistical sampling method for calibration. The TDC is programmed entirely on-chip, digitally. It measures timing slack in a CPU that continuously works to maximize its clock frequency while maintaining 0 timing failures. This all-digital TDC and calibration scheme fits within the context of a digital processor that is constantly monitored on a cycle-to-cycle basis to maximize the clock frequency while avoiding failure. The TDC is discussed in chapter 2, and is based on the work in [7].

In chapter 3, we present a novel way of exploiting the variation inherent in the oxide breakdown process, in order to generate permanent, on-chip IDs. We present dynamic, run-time adaptive algorithms that automatically calibrate the on-chip circuitry and controller to compensate and adjust for the oxide breakdown process in order to generate highly random IDs that are stable. These IDs are useful in high security applications, such as those that require guaranteed permanence of fingerprinting a chip. The system we present is called OxID and is based on the previous work we presented [8].

In chapter 4, we further extend the theme of dynamic run-time calibration algorithms for tuning on-chip circuitry, but in this case, to optimally exploit the time to oxide breakdown for generating true random numbers in a system we call OxiGen. This true random number generator is based on a novel physical randomness source: time to

oxide breakdown. We show that this is the first on-chip true hardware random number generator to pass *all* fifteen NIST 800-22 randomness tests, without the aid of a post-processor. The work presented in this chapter is based on the work we presented in [9].

Finally, in chapter 5, we explore variation-robust and variation-minimizing circuit design techniques for operation in subthreshold while maintaining comparable performance to more traditional CMOS circuits. We apply these techniques to an on-chip implementation of a self-timed transition detection circuit that takes in many inputs, fully-robust to variability in all corners and Monte Carlo runs. The self-timed transition detection circuit is thus a fully automated self-tuning circuit. The transition detection circuit was implemented in the Razor chip of chapter 2 [7], and a stand-alone core of the Phoenix 1 processor [10].

# CHAPTER II

## A 5 ps Time to Digital Converter for Measuring Timing in a Razor-type System

In this chapter, we explore the measurement of variability due to process, supply, and temperature, by using highly accurate digitally calibrated and digitally designed time-measuring circuits. PVT effects and variation all ultimately surface as timing delays through a logic block, and by monitoring the critical paths of a design, delays caused by PVT variation can be measured and properly accounted for. The Razor-- system design (otherwise known as “Safety Razor”) plans to do just this: monitor timing signals of critical paths accurately, and thus be able to store and pre-calculate the margins required by the system to correct for global and random variation, temperature, and aging effects (such as NBTI and TDDB). The key component of the Razor-- system is the highly accurate, statistically calibrated time to digital converter (TDC). In this chapter, we delve into the origins, novelties, design decisions, and design techniques employed to design such a TDC.

## 2.1 Introduction to time to digital conversion

Time to digital conversion is the process of measuring, on-chip, the time elapsed between two signals, and outputting the measured time as a digital signal or codeword. Usually, the two signals are a start and data signal. There are several basic types of time to digital converters (TDCs), which we will elaborate on below; most TDCs are based upon similar basic circuit architecture, with more complicated ones using more complicated blocks that replace simpler blocks, or that add extra logic.

The most basic TDC architecture, and one which is foundational to most other TDCs, is a series of flip flops clocked by a delayed start signal, with their data inputs being fed by the same data signal. This basic architecture is illustrated in Figure 2.1. The start signal is delayed by an appropriately sized buffer delay alpha, and is delayed by alpha from flop to flop. Thus, each flop ***i*** effectively samples the data signal at time *i*alpha*; if the input data signal is a step waveform, the digital codeword that is output by the series of flip flops will be a thermometer code that has a pattern of all 1's to 0's or all 0's to 1's.

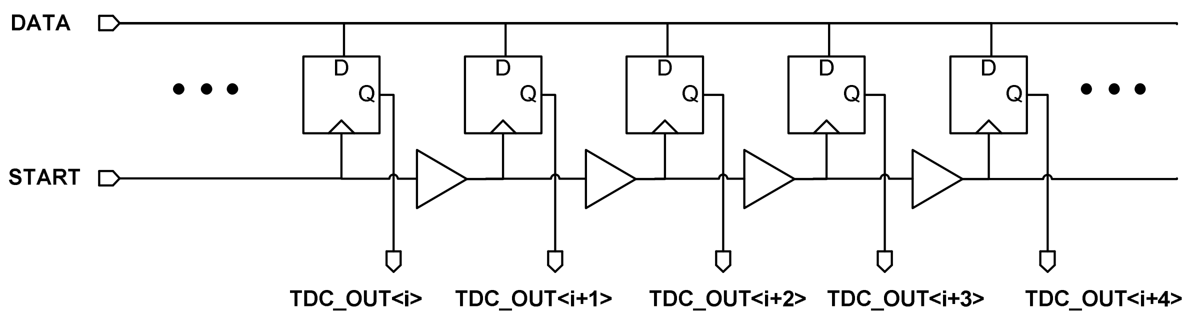



**Figure 2.1: Basic Start-Delayed TDC**

Similarly, as shown in Figure 2.2 below, one can delay the data signal instead of delaying the start signal. Sampling the data signal instead thus samples delayed versions of the data signal (alpha*1, alpha*2, alpha*3, …) at the time the start signal switches. This is effectively the same as the above, but the codeword is instead flipped backwards in comparison to the above architecture. This flipping occurs since the data signal at the end of the chain is alpha*n backwards in time (the oldest signal), while the first data signal is alpha*1 (newest signal) in time.

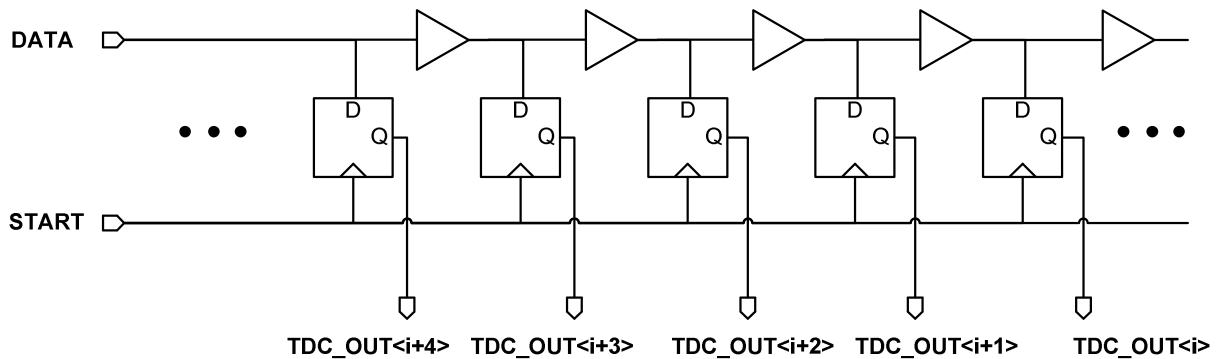



**Figure 2.2: Basic Data-Delayed TDC**

If the smallest digital buffer delay in a process is alpha, it possible to measure delay resolutions smaller than alpha. By delaying both the start and data signal, by alpha+d and alpha respectively, one can measure resolutions of (alpha+d)-alpha = d. If one delays in reverse, such as delaying the start and data signal by alpha and alpha+d respectively, then one can measure resolutions of d with the codeword flipped backwards in time. Figure 2.3 illustrates this third basic architecture.

Figure 2.3: Vernier start-delayed TDC (top); Vernier data-delayed TDC (bottom)

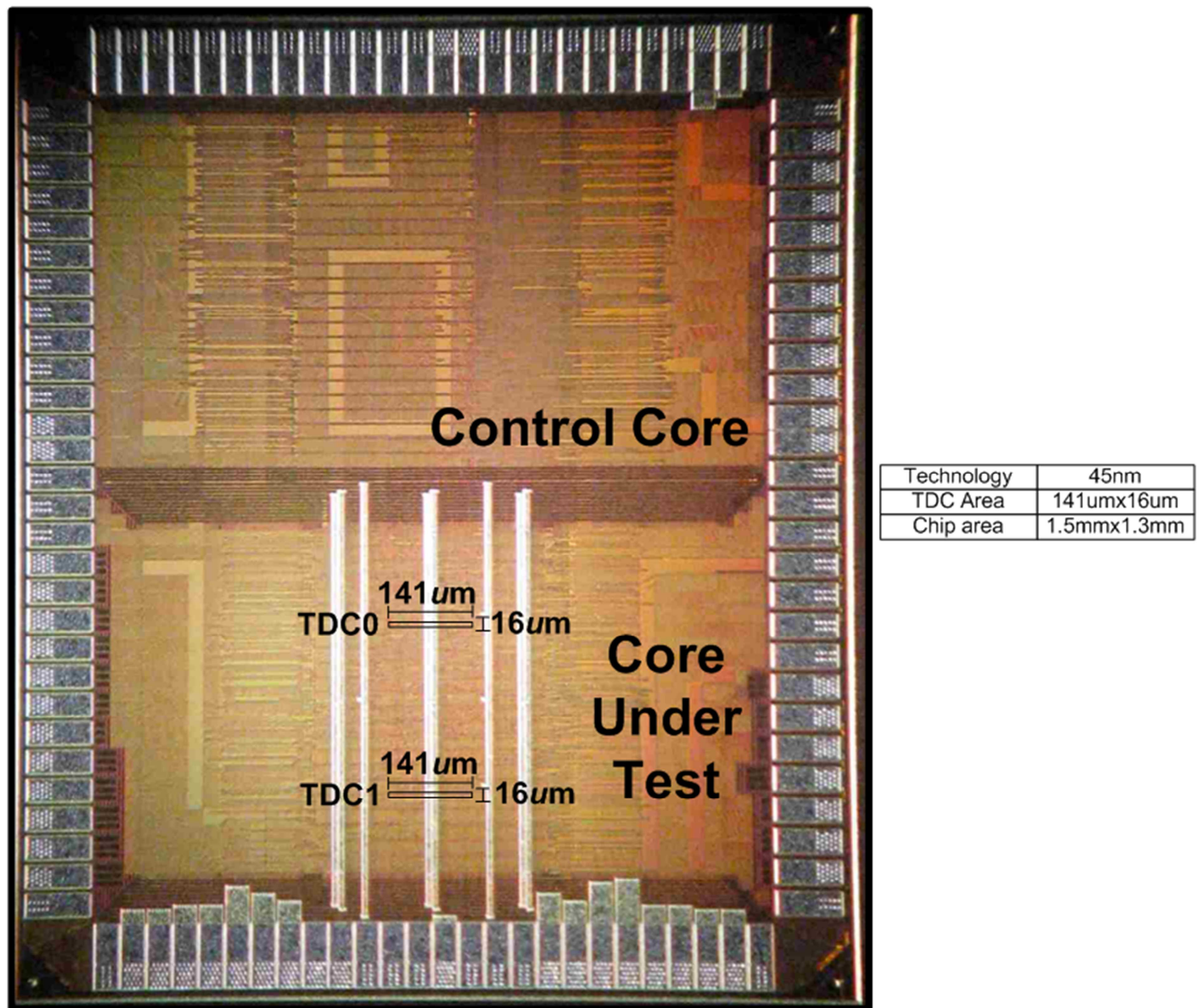
However, this configuration is hard to achieve in reality, for a resolution *d* that is much smaller than the basic buffer delay of the process. This is because process variation, mainly, mismatch, will cause each stage to be skewed. Given enough stages, this skew can accumulate to unmanageable proportions, given more worst-case scenarios. With even just a few successive stages skewed, the TDC measurement is effectively crippled, since later stages are then offset by delays that are unknown and cannot be measured without extra circuitry. Even worse, these cases change from die to die and the TDC measurement is effectively unusable.

Therefore, a primary challenge of using this third basic architecture is designing a calibration scheme and circuitry that can either calibrate or cancel out the mismatch introduced by process variation. Calibration requires a one-time effort, and the calibrated results can then be saved. The disadvantage of calibration is that each TDC on each die must be individually calibrated, a process that could prove costly and time consuming. In addition, circuitry that cancels mismatch also adds extra penalties of control, power, and area overhead.

## 2.2 Concepts of proposed time to digital converter

Advanced CMOS technologies have become highly susceptible to process, voltage, and temperature (PVT) variation. The standard approach for addressing this issue is to increase timing margin at the expense of power and performance. One approach to reclaim these losses relies on canary circuits [11] or sensors [12], which are simple to implement but cannot account for local variations. A more recent approach, called Razor, uses delay speculation coupled with error detection and correction to remove all margins but also imposes significant design complexity [13, 14]. In this chapter, we present a minimally-invasive in situ delay slack monitor that directly measures the timing margins on critical timing signals, allowing margins due to both global and local PVT variations to be removed.

Figure 2.4 shows the implementation of the proposed slack monitor in an Alpha processor fabricated in a 45nm process. The core interacts with the TDCs through memory mapped IO.




| Technology | 45nm |
|---|---|
| TDC Area | 141umx16um |
| Chip area | 1.5mmx1.3mm |

**Figure 2.4: TDC Die photo in addition to test core and configuration core.**

In Figure 2.5 we analyze the accuracy of the RDC statistical sampling method. After setting the RDC to a fixed phase pulse width (min, middle, max), we swept a range of calibration counter values and plotted the standard deviation of the reference counter after the calibration counter reaches its target value; trials continued until the calibration counter reached a particular value and then stopped, resulting in a corresponding

reference counter value. The stability of readings indicates a deviation below 1ps for a calibration counter of 500,000 samples. Additional accuracy can be gained with exponentially larger calibration counter values, or by using stronger statistical methods than averaging. In this implementation, the reference clock was implemented with a current starved ring oscillator; a crystal oscillator would provide further accuracy.

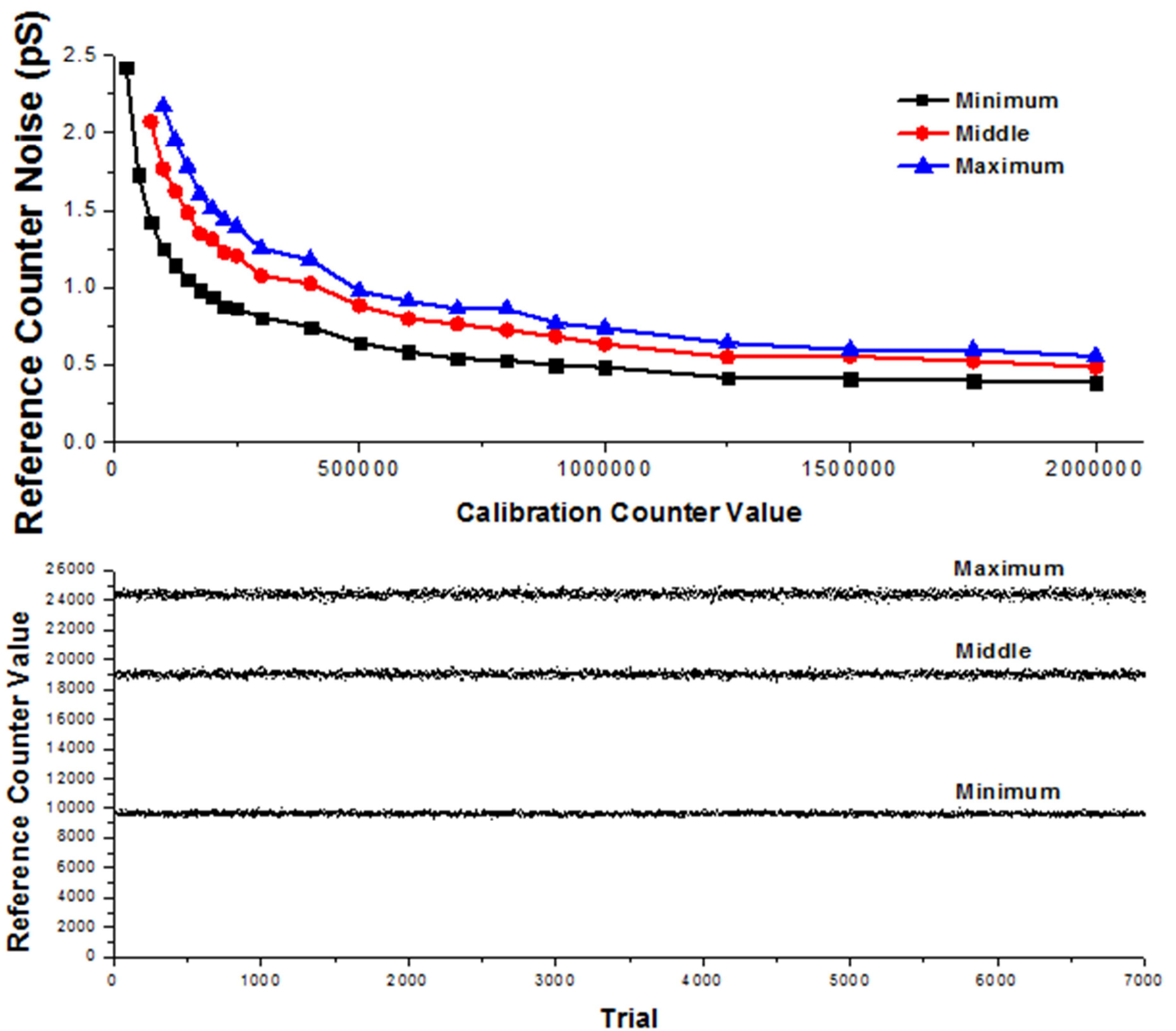



**Figure 2.5: Reference counter noise vs count value (top); reference count vs trial number.**

## 2.3 Circuit implementation of time to digital converter

At the heart of the monitor is an all-digital, self-calibrating time-to-digital converter (TDC) consisting of a 30-bit Vernier chain (VC) with each stage tuned to 5ps increments. The overall measurement window is 150ps, which is sufficient for timing slack measurements in 2 Ghz+ processor systems. A tunable delay chain is also included to allow a wider range of measurements, if necessary. A key concern in TDCs is the need for time consuming and complex calibration, which is especially difficult given the very high resolution demanded by high-performance processors. To avoid expensive off-chip measurements and tester procedures, we propose a new approach that allows the TDC to be automatically self-calibrated under the sole control of the processor using only the off-chip crystal oscillator. We use a process of statistical sampling to convert the relatively slow system clock to 1ps accurate calibration steps. The proposed approach was implemented on a 64-bit Alpha processor and can complete full self-calibration of the entire TDC in five minutes with approximately 1ps accuracy. Calibration can be performed in the field during idle periods, incurring no additional tester time or system downtime. Typical delay slack margins remain in place until the TDC is calibrated.

The TDC architecture is shown in Figure 2.6. Figure 2.7 shows the layout of the TDC; note that the high-metal power straps and decap are not shown, for clarity.

**Figure 2.6: TDC architecture and schematic design of customized vernier delay element.**

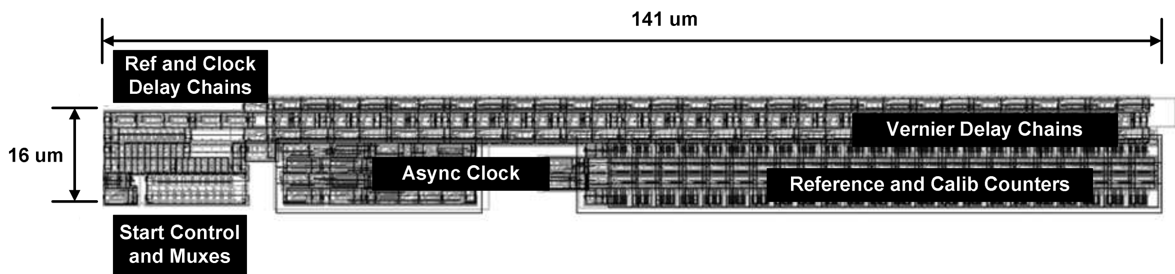



**Figure 2.7: TDC layout without high-metal power straps and decap (omitted for clarity).**

A measurement is initiated using the start signal, which allows the next clock edge to pass through a latch and into the device. The clock passes through the tunable delay chain, which allows coarse alignment of the detection window. The VC is calibrated so that each stage delays the clock by 5ps relative to the data, resulting in a forward sampling of time. The clock then exits the VC and triggers a TDC done signal, which requests that the processor read the latch states. Each delay element in the VC is tunable using eight identical processor-controlled capacitor loads, which are designed to induce 1ps shifts in delay. A uniform load structure was chosen in lieu of a more compact binary weighted scheme to ensure monotonicity, which greatly eases automation of calibration.  Narrow pulses (glitches) can occur on data signals and must be monitored since they could be captured by a pipeline register if delay slack margins are reduced too far. To prevent pulses from collapsing in the VC due to rise-time/fall-time imbalance, the chain lengthens pulses by accelerating falling transitions forward using speedup signals as shown in Fig. 2.6. Because of this, the TDC is calibrated for rising transitions - we include an XOR gate in order to monitor falling transitions. The metric of interest is the slack between the latest arriving input signal and the clock edge at the flip-flop (FF) input. To compute this metric, a mux local to the monitored FF selects between clock and data, and we subtract the two measured delay values. The number of FFs that need to be monitored depends on the balance of the pipeline; typically 30% of all FFs is sufficient [13].

## 2.4 Circuit techniques used for improving time to digital converter accuracy

The detailed circuit schematics for the TDC implementation are shown in Figure 2.8.

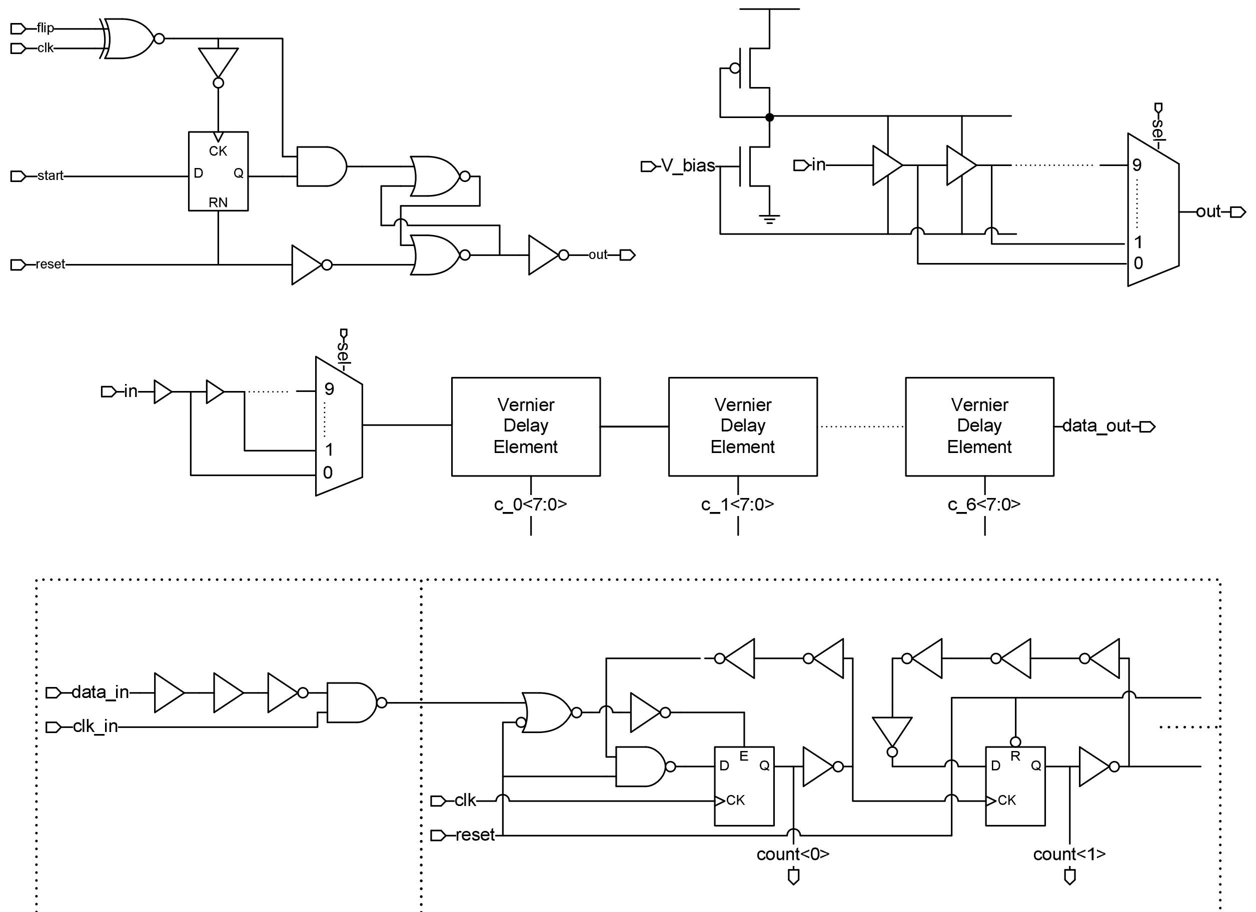



**Figure 2.8: Circuit implementations of key components of the TDC block.**

In this section we will discuss the particular circuit optimizations and techniques used to maximize the resolution accuracy and precision of the TDC; in particular, the design of the vernier delay chain. To ensure that the single-weighted capacitors on each

node of the delay chain can give us our required tuning range, we first designed their number so that there was sufficient overlap amongst the delay they added, for full mismatch variation including both the capacitors and the buffers. Thus, we found that, over all worst-case monte carlo corners, eight capacitors per node were sufficient. In addition, the size of the capacitance (mosfet with source and drain tied) chosen was important, since a small size requires more caps (and thus much more control and area overhead), while a large size limits resolution. The size was chosen such that, again, across all worst-case mismatch variation, there was sufficient overlap amongst all the capacitors per node to give the required combination of total capacitance per node to achieve the target resolution.

## 2.5 TDC calibration scheme

The VC is calibrated using a reference data transition (ref_tran) generated by the reference delay chain (RDC in Fig. 2.6). The RDC is trigged by the clock and feeds test transitions into the data input of the VC. First, we find the zero point of the RDC (which is tunable in 1ps increments) by increasing delay until the zeroth bit of the VC shows equal probability of reading zero and one. The goal is then to shift ref_tran in exact increments of 5ps and tune the subsequent bits of the VC in the same manner. To accurately measure shifts in ref_tran under PVT variation, a measureable reference pulse (ref_pulse) is generated by subtracting clock and ref_tran. The signal ref_pulse is intentionally lengthened to ensure non-zero pulses when ref_tran and clock are aligned. As explained below, the length of ref_pulse is computed by comparing it to a known

calibration pulse (cal_pulse) using statistical sampling, which is generated using a reference off-chip crystal. Although the exact relationship between the length of ref_pulse and the alignment of ref_tran cannot be accurately determined, it is unimportant since we require only that a 5ps shift in ref_tran will result in an identical 5ps shift of the ref_pulse length. Hence, we search for an RDC setting that produces an exact 5ps increase in ref_pulse. This ensures that ref_tran is delayed by an exact 5ps step and allows the next bit of the VC to be tuned.

To determine the length of ref_pulse, a start signal generates a single ref_pulse and cal_pulse, which are each fed to the enable input of identical up counters. A slow asynchronous sampling clock generated locally provides the clock input for both counters. Since it has no relationship to either pulse source, it statistically samples both signals over many trials (1M in our tests). This technique allows us to compare the length of ref_pulse relative to cal_pulse with sub-1ps accuracy despite pulse widths of hundreds of ps. Calibration error is not cumulative as the process moves through the VC since each ref_pulse is compared to the length of the zeroth-bit pulse.

## 2.6 Silicon implementation and measurement results

In Figure 2.9 we show the calibration settings of a VC delay chain. The “skew” signal for the clock chain side is enabled to provide a baseline delay of 5ps. To address local variation, the 1ps calibration capacitors are enabled as needed on either chain to obtain equal probability of a zero or one reading when the reference delay chain is stepped with 5ps increments. In this experiment we ran 10,000 trials to check the 50%

probability condition (corresponding to a count of 5000) for each bit in the VC chain. Each time the condition is checked we try to find an improvement by enabling another capacitor on one side of the chain. For instance, if a count of 8000 is recorded, then the data transition is coming before the clock transition too frequently, so we enable a data capacitor to try to reduce the count. Once there are no more capacitors available, the configuration that yielded the best 50% condition is restored and the next bit of the VC is calibrated.

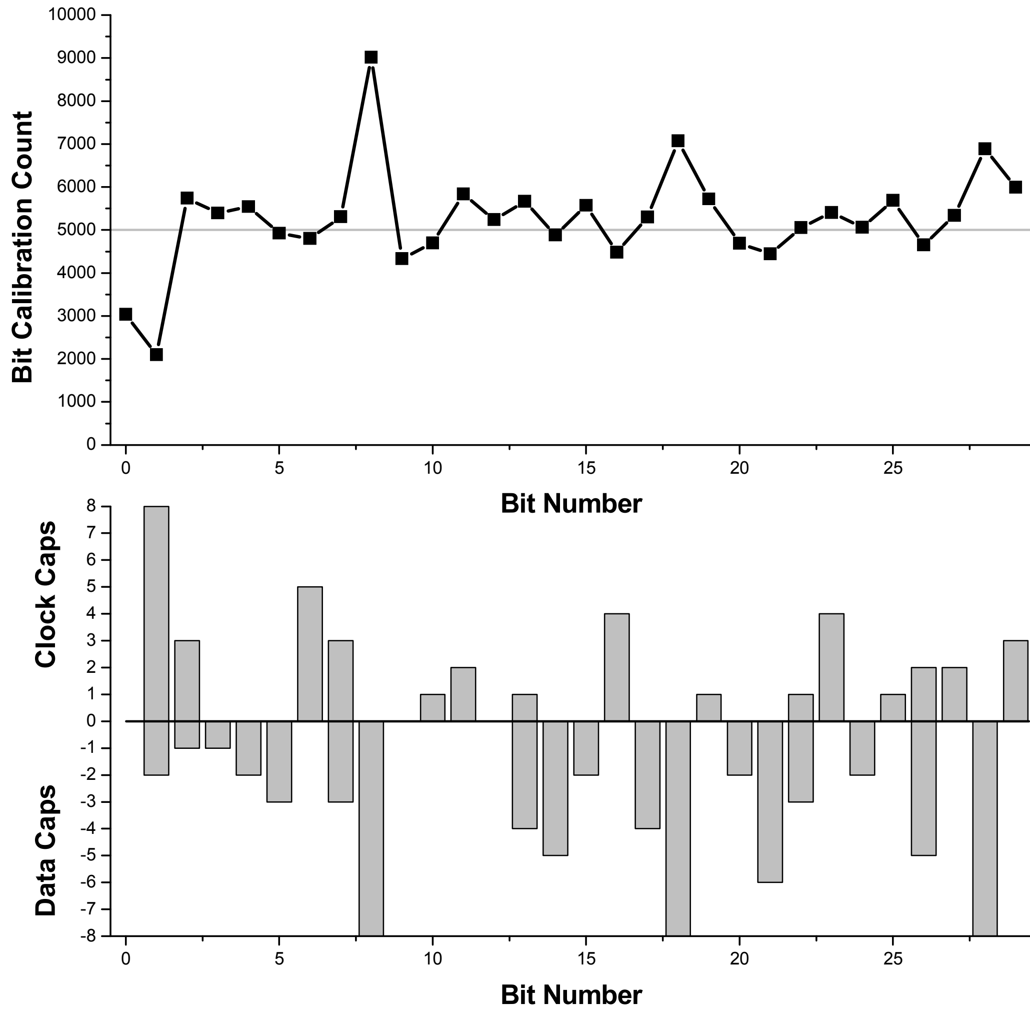
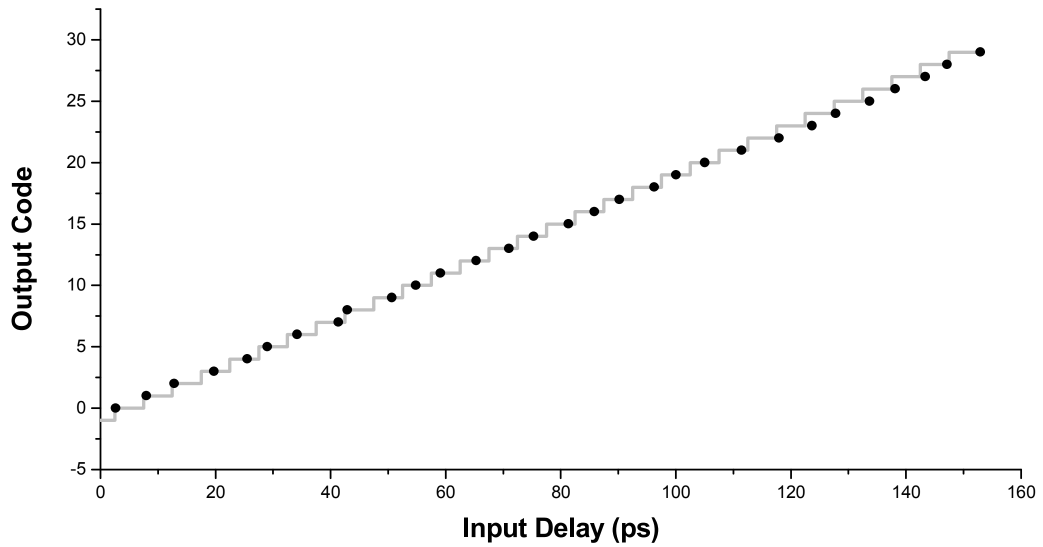



**Figure 2.9: Recorded bit counts for each TDC element (top); corresponding data and clock caps used for each bit.**

In Figure 2.10 we use the reference delay chain to sweep 1ps step inputs through the VC and show output codes. Monotonicity is guaranteed due to the calibration method and also shows excellent linearity.



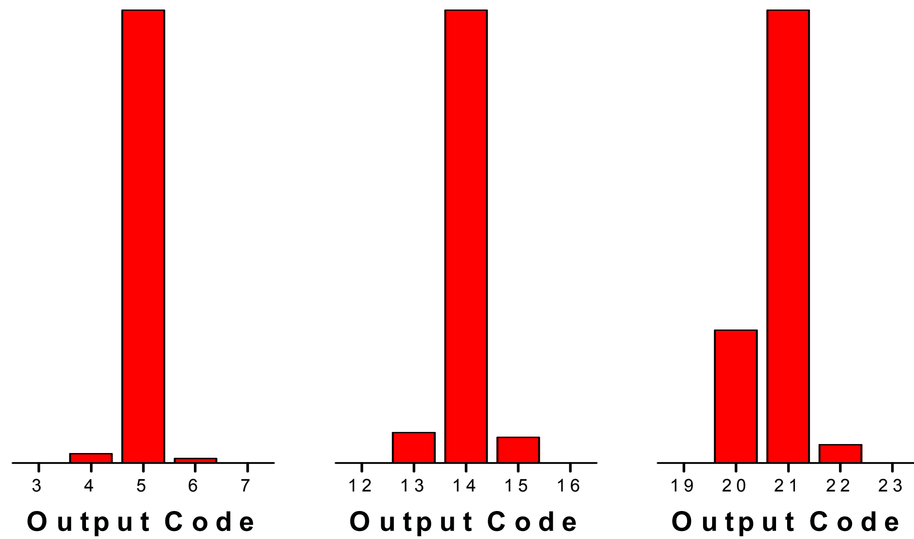




**Figure 2.10: Output codes corresponding to input delay using reference delay chain (top);Histogram of output codes showing low noise.**

Finally, the use of the TDC is illustrated by using it to measure a timing slack distribution of the most critical paths in the Alpha processor, as shown in Fig. 2.11, illustrating the potential use of the TDC.

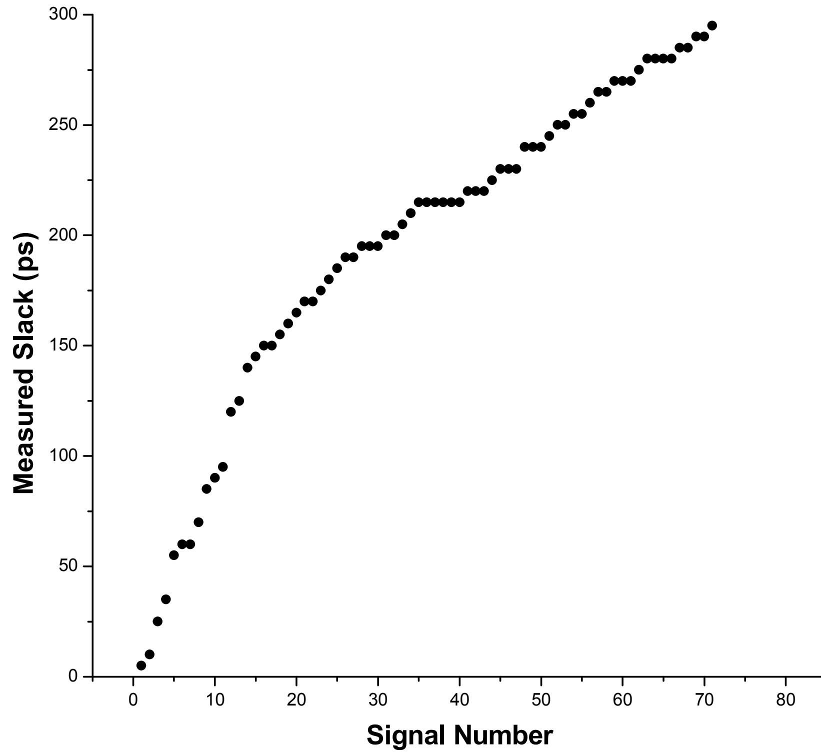



**Figure 2.11: Slack of Alpha core critical paths measured by the TDC.**

## 2.7 Summary and discussion

The TDC presented above is all-digital, highly accurate, and relatively easy to calibrate through the programmable second programming core. It plays a critical role in a Razor [13, 14] type system where the core is running constantly at its maximum frequency, by monitoring itself to adapt to PVT conditions and PVT variation. If it is pre-calibrated for all PVT corners, and by storing its digital programming values, it can theoretically monitor the system on a near cycle-to-cycle basis (minus the few pipeline

cycles it takes for the instruction that programs it to complete given that new PVT conditions have been sensed). Once new PVT conditions have been sensed, it can then reprogram itself with the right instruction (a few cycles of latency for the instruction). The system latency in adapting to the new critical path delay sensed (ie maximum clock frequency) is not dependent upon the TDC, but instead dependent upon the clocking circuitry used to generate the on-chip clock.

In future revisions and versions of this TDC, research can investigate how the TDC range can be increased for cores ranging widely in supply voltage (ie operable from subthreshold to nearthreshold to superthreshold). Or, more intelligently, research can examine methods for allowing the TDC's buffer delays to scale with voltage, either with or without the help of some digital or analog trimming or biasing. The range can be extended via some type of coarse-fine architecture, as well.

# CHAPTER III

## OxID: On-chip One-time Random ID Generation using Oxide Breakdown

This chapter explores a particular application of leveraging device exploitation to our benefit. We examine the application of oxide breakdown for on-chip ID systems. As oxides are subject to variations in their thickness (on the order of a few single layers of atoms in the latest processes), the circuit and system must be able to calibrate to this variation, as it shifts the mean of the oxide breakdown time distribution from chip to chip. This chapter aptly demonstrates the use of automated circuit tuning in order to overcome this variation and achieve high quality results that are otherwise extremely difficult to achieve. In addition, the system presented is the first known use of oxide breakdown to generate chip IDs; the presented chip ID system has highly random IDs that are highly stable, both very crucial qualities in a chip ID system. By tuning the circuit and adapting to the oxide thickness variation for optimal results, we are in fact exploiting this on-chip variation for our benefit. Thus far, on-chip variation exploitation has been limited to a specific few digital applications, such as on-chip ID generation and on-chip random number generation. Numerous other broader classes of applications exist that benefit greatly from variation exploitation: frequency and power binning to enhance

yield and increase profit; scientific simulations; cryptography; and high-level security applications that require randomness in ways other than random numbers. Variation continues to increase as processes scale further, and as this increased variation is unavoidable, we should find ways to automatically tune our circuits to exploit it to our benefit optimally, in addition to solely fighting the one-way uphill battle of mitigating it.

## 3.1 Introduction to oxide breakdown

Gate oxide breakdown begins when an extremely high voltage (and thus an extremely high electric field) is put on the gate of a mosfet. When this occurs, current tunnels through the oxide, and depending upon the oxide thickness, either of two current tunneling mechanisms can occur. For thick oxides [15, 16], the gate tunneling current mechanism is Fowler-Nordheim tunneling, while for thin oxides less than 3 nm (30 angstroms), and for voltages less than 3V, the gate tunneling current mechanism is due to direct quantum mechanical tunneling. As current tunnels through the oxide, defects are generated within the oxide, causing local electric fields to be formed. These defects and their local electric fields further increase the defect generation rate as current tunnels through, and more, new, defects are formed at faster and faster rates. Thus, a positive feedback cycle is formed and eventually, the gate oxide reaches a state where it has destructively broken down (the defects reach a critical density); at this point, the gate oxide stops functioning as an insulator and behaves similarly to a resistor.

## 3.2 Introduction to on-chip chip-ID systems

Chip ID systems are used in enforcement of user licenses as well as in communication and security protocols. In these applications, it is desirable to generate IDs on-chip at the application point so IDs are guaranteed unknown until first used. This avoids the need for off-chip, pre-generated IDs that are programmed using fuses, a process that exposes IDs to human intervention and storage on computers that may be compromised.

A key requirement for chip ID generation is that the generated ID is unique to only that chip, and that the ID is time and environmentally invariant. The chances that two chip IDs have all, or many, bits the same is minimized by using a large bit width (e.g., 128bits/ID) and ensuring a high degree of randomness during generation. Thus, for a chip ID system to be useful, IDs should satisfy two main properties: that no two IDs are the same, and that each ID is time and environmentally invariant. The former means that no two chips can be mistakenly identified to be the same, while the latter ensures absolute reliability when reading IDs; that is, the ID for each chip will not change over time or environment. The Hamming distance metric is used to measure the degree to which a chip ID system matches these ideal properties: the Hamming distance measures the number of differing bits between two same-length bitstrings. Therefore, for the first requirement that no two IDs be the same, the Hamming distance should be ideally half the total number of bits in the ID bitstring. For the second requirement that each ID can be stably read, the Hamming distance is measured with respect to the same ID, thus called the self-Hamming distance. The self-Hamming distance ideally should be 0 for every ID.

Previous methods rely on inherent threshold voltage (Vt) mismatch between devices, which is detected by measuring either device current [17] or inherent SRAM bitcell skew towards 0 or 1 states [18]. However, Vt mismatch can be very small between any particular transistor pair, making it difficult to repeatedly generate an identical ID for a given chip. Hence, previous approaches exhibit a small number of bit flips between successive ID readings (i.e., the IDs had a non-zero self-Hamming distance), complicating the use and reliability of chip IDs.  We discuss these previous methods in detail as follows.

The first known on-chip generated chip ID system was the work presented in [17]. Fundamentally, it presented an array of mosfets whose currents will vary due to mismatch, which is random due to random dopant fluctuation; this array has common gate and source, but the drains are selectively muxed into the same resistive load to generate voltage drops proportional to their current. Thus, the voltage drops will vary randomly as the current also varies randomly with mismatch. The system, called ICID, takes these sequences of random voltages and an auto-zeroing comparator, and successively compares these voltages against each other to translate them to a continuous bitstream of 0's and 1's. The system, with ID bit length of 112 bits, was able to generate a large set of IDs with minimal collisions, but does have bit changes "typically less than 5%."  In addition, the system acknowledges the effects of threshold voltage shift over time due to aging effects as well as ID drifts due to high temperature.

The other known on-chip generated chip ID system is that of [18]. It presented a SRAM type memory array of 128 bits, representative of a 128-bit ID. Each SRAM type cell is a cross-coupled NOR-gate, with both sides having an input of the NOR as a shared

reset. Thus, upon release of reset, the cross-coupled cells will fall towards either 0 or 1, depending upon the mismatch in Vth. The work emphasizes that, as it is based upon the cross-coupled positive feedback of the latch, it does not need to rely on "offset-nulled comparator or low-offset amplifier"; that is, it "[does not need to] detect very small Vt variations." The work demonstrates achieving high-quality Hamming distances of 64.16, but does exhibit "3.89 unstable bits per [ID]."

## 3.3 On-Chip vs Off-Chip ID Generation

At this point, it can be asked: why bother with on-chip ID generation, which has to meet the aforementioned requirements, if one can simply generate the ID off-chip and program it onto the chip using on-chip fuses? This question brings us to the discussion of on-chip vs off-chip ID generation. Clearly, there are two main methods for generating chip IDs: on-chip using circuitry that is able to generate randomized IDs, and off-chip generation using software or another hardware generator, and then subsequent programming onto the chip via on-chip fuses.

Direct, on-chip generation has numerous security advantages over off-chip generation, since it eliminates many human-involved steps that expose the ID to potential tampering and snooping. For example, off-chip generation requires using software to generate the IDs, a possibly error-prone or even malicious process. If off-chip generation involves other hardware generator chips, rather than software, there is still at least several different, physical points that the resultant ID must be passed through, translating to insecurity. Off-chip generation, no matter using software or other discrete hardware,

requires the ID to pass through different steps in a flow, thus exposing the ID to malicious snooping or tampering. Direct, on-chip generation by its inherent nature completely avoids these problems: the ID is generated in a single-step on-chip, eliminating ID exposure and the potential for human-tampering.

Clearly, direct on-chip ID generation is more secure, by wholly sidestepping the numerous security flaws of off-chip generation. However, far steeper requirements are required to reap the benefits of on-chip generation: it is difficult to ensure *a priori* that any two IDs will be different, and that each ID reads exactly the same (no bit-flips) under process, voltage, temperature, and time.

## 3.4 Concepts of proposed OxID on-chip chip-ID system

We present a new method called OxID that generates chip IDs using oxide breakdown. We leverage the fact that oxide breakdown is an inherently random effect [19] (one oxide may break long before another identical oxide under the same stress conditions) and is also both abrupt and permanent. Hence, it enables improved ID stability over time and environmental conditions. Once an oxide breaks down, its resistance changes from a nearly infinite value to the order of MΩ or kΩ [20], which has made it popular for one-time-programmable arrays [21,22]. Silicon measurements of 162 ID generators in this work demonstrate nearly ideal randomness of the generated IDs, maximizing their uniqueness. The proposed approach can also detect prior ID generation; if on first use the ID is non-zero, this indicates that the ID was previously generated through possible intrusion and may be compromised.

OxID consists of a memory array composed of 3-T memory cells that use a thin-oxide moscap as a fuse element (Figs. 3.1 and 3.2). The array has 16 rows by 8 columns, totaling 128 cells, each of which can be read through a bitline and sense amplifier.

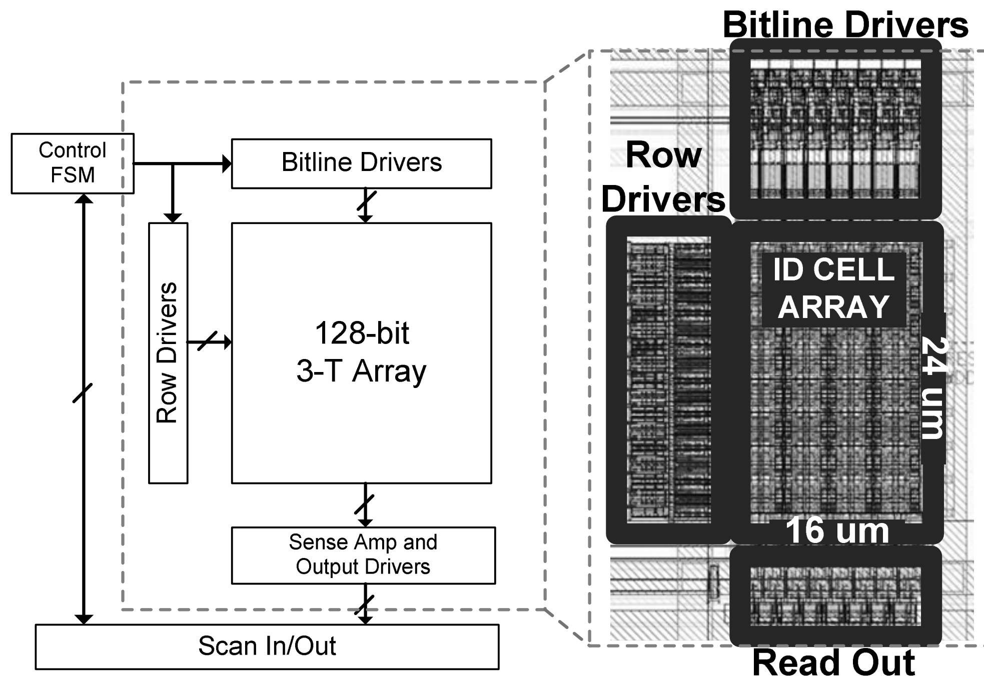



**Figure 3.1: System architecture with individual array layout.**

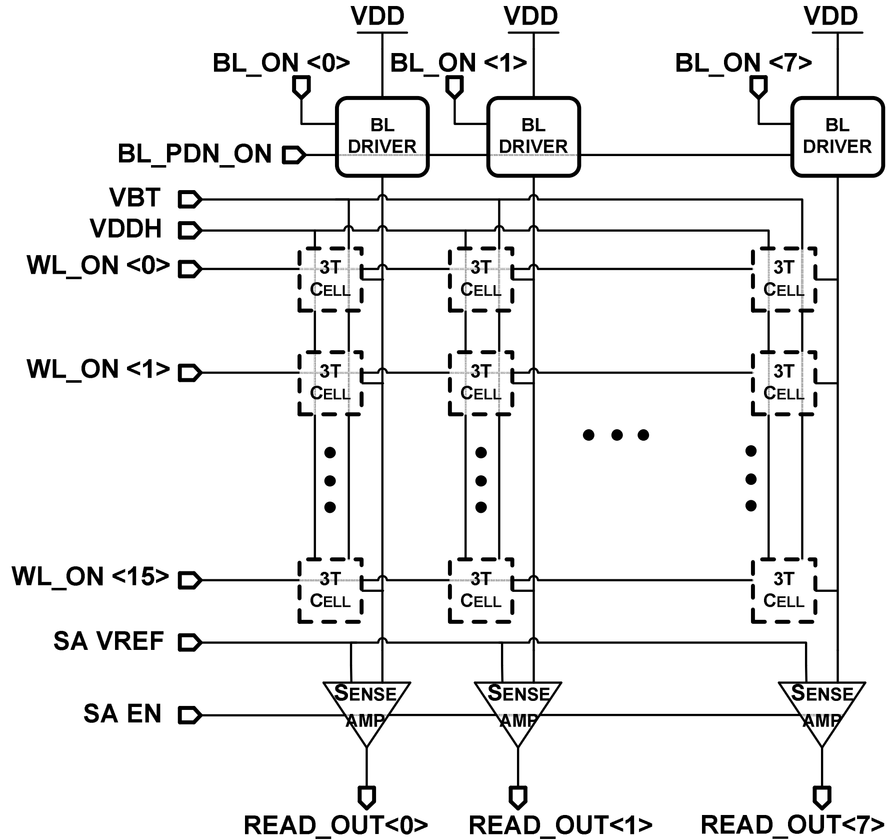



**Figure 3.2: Schematic architecture of a single array.**

All oxides in the array are exposed to a stress voltage of 4.5V and identical stress time. While the breakdown times of the 128 oxides follow a random distribution, the mean of this distribution is difficult to determine a priori and can change from chip to chip due to oxide thickness variations. Hence, exposing all chips to a preset stress time will likely result in a significant portion of OxIDs with oxides either all broken or unbroken. Therefore, we propose two algorithms that dynamically adjust stress time to ensure that close to half of all oxides break while half remain intact. Both algorithms stress the array in small time increments using an on-chip controller. In the first algorithm, the entire array is read out after each stress interval. Initially, the array will

read nearly all zeros and gradually contain more ones as oxides start to fail. When ones exceed zeros, the stress iterations are terminated and the ID is complete. By dynamically checking the array state after each stress interval, the algorithm automatically adapts to the global condition of the oxides, providing added stress to more inherently reliable arrays. It also provides immunity to voltage fluctuations during the stressing.

One drawback of this approach is that all generated IDs will have a nearly identical number of zeros and ones. While this does not reduce the randomness of the IDs, it does reduce the number of possible ID permutations. For a 128b ID, the number of possible ID permutations reduces by a factor of ~23.8. Hence, if an equivalent pool of IDs is required as in a standard random ID, the number of bits must be increased (e.g., for a 128b ID, by 4 bits or ~3%). Therefore, we propose a second algorithm that uses a small set of canary cells to predict the number of stress iterations for the entire array. In this case, only cells specified as canary cells are read out after each stress interval and further array stress is terminated when 50% of the canary cells are broken. Due to random variation between the canary cells and the remainder of the array, a larger se of ID permutations is generated.

Both algorithms are implemented and compared. After the ID is generated using either algorithm, a final "afterburn" phase is performed where all broken oxides are strongly stressed for a longer duration. Due to limitations of stress isolation a few borderline oxides may break down as well. Hence, this process sacrifices a small Hamming distance degradation (measured at 2-3%) for higher read operation robustness across environmental conditions.

## 3.5 Implementation of OxID circuit and system

The 3-T bitcell (Fig. 3.3) consists of a thin-oxide SVT transistor driven by the wordline, a thick-oxide 2.5V I/O “blocking” transistor and a thin-oxide SVT transistor with S/D tied as a moscap. This bitcell is similar to the 3-T cell in [20,23]. The thick-oxide transistor separates the thin-oxide wordline device from the high voltage of the moscap during stress. For unbroken oxides there is by design a small voltage that accumulates across the moscap due to its high leakage a high $V_{DD}$ (0.7V for $V_{DDH}$ of 4.5V). This protects oxides that have not been selected for stress. Cell currents are limited by the resistance of the minimum-sized thick-oxide transistor and word-access transistor. During cell read, $V_{DDH}$ is shorted to $V_{DD}$. For experimentation, the 128 oxides can be stressed all at once or by row, column, or cell. Figure 3.4 shows the system layout for a single core; there were two cores implemented for the chip.

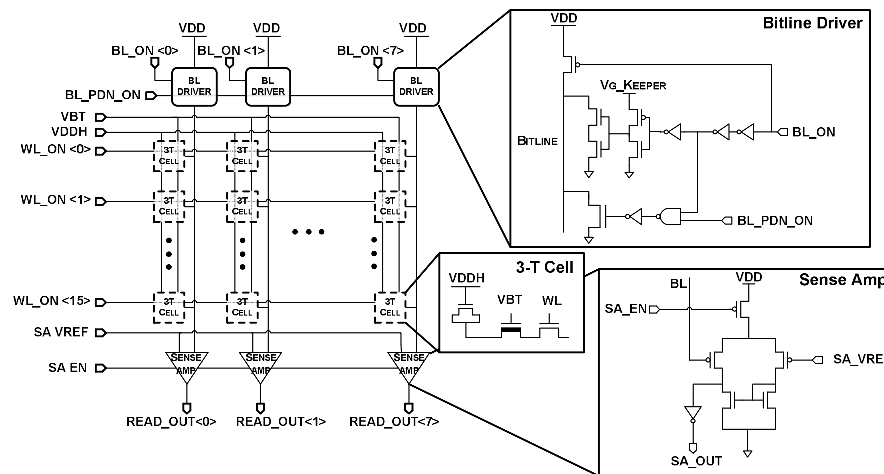



**Figure 3.3: Schematic architecture of a single array with schematics of key components.**

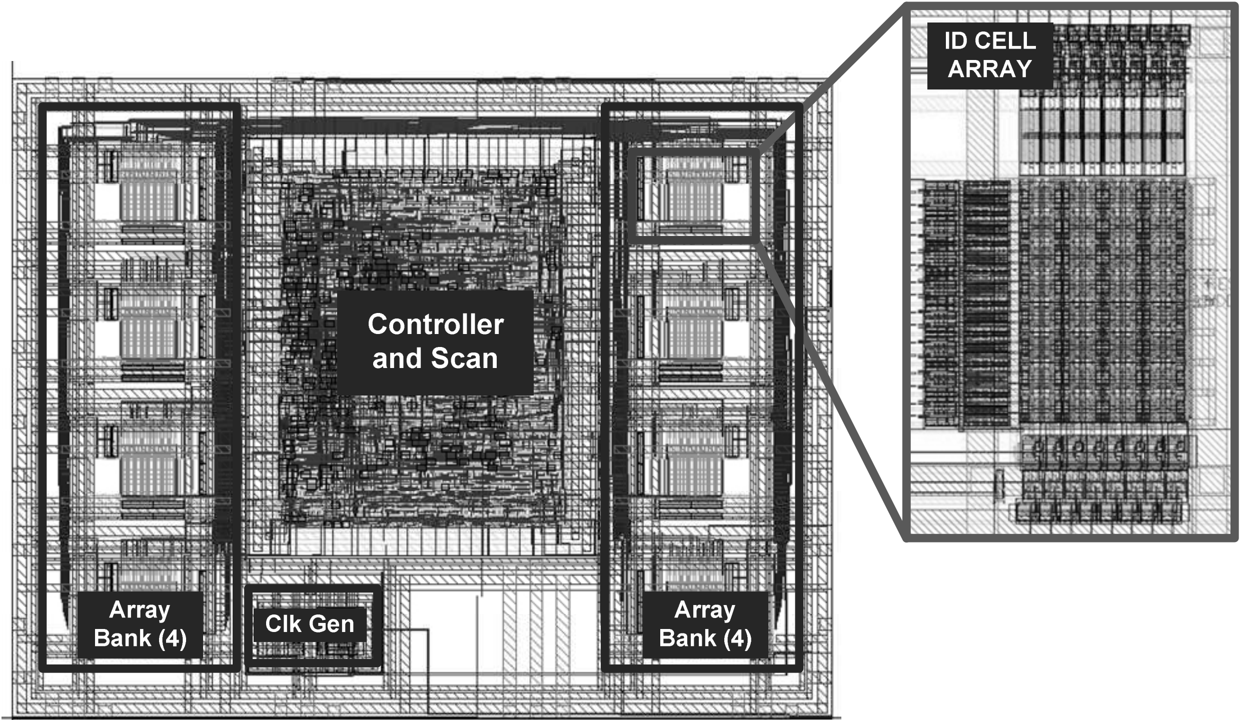



**Figure 3.4: Physical layout for a single core.**

## 3.6 Measurement results and discussion

OxID was implemented in a standard 65nm CMOS technology. For experimentation, the gate voltage for the blocking transistor (VBT) and the sense amplifier reference voltage were brought in from off-chip, but can also be generated on-chip. We applied the global stress algorithm described above at room temperature to 162 arrays and the canary-based algorithm to 144 arrays. Two perfectly random IDs should, on average, have a Hamming distance of exactly half the total number of bits in the ID. Comparing all pairs of ID bit sequences (13041 and 10296 pairs, respectively), the average Hamming distance for the global algorithm is 63.92, close to the ideal value of 64 (Fig 3.4). The average Hamming distance for the canary algorithm is 61.79, implying a trade-off in randomness and ID set size (Fig 3.5).

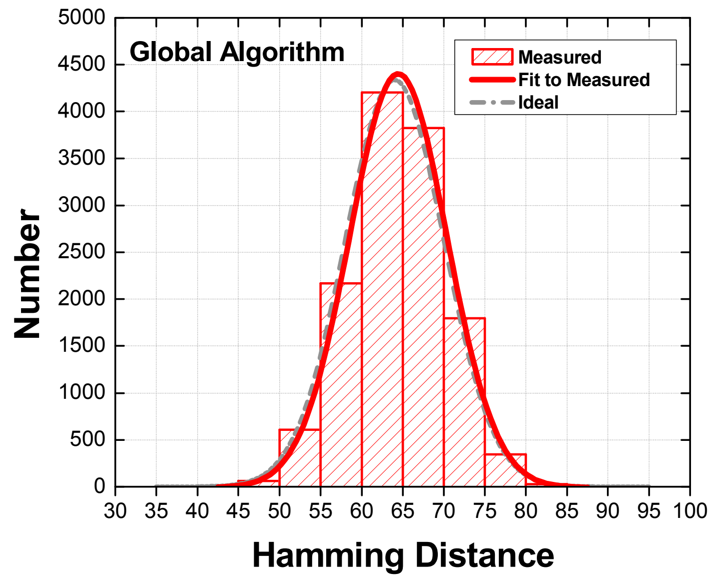



**Figure 3.5: Hamming distance results for global algorithm.**

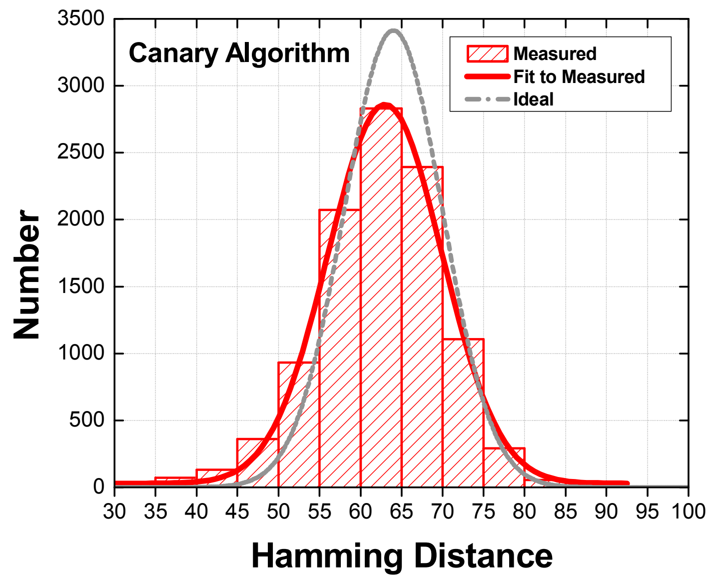



**Figure 3.6: Hamming distance results for canary algorithm.**

The self-Hamming distance upon repeated reading of the ID in different environmental conditions was tested for 14 arrays. Results show 0 self-Hamming distance for up to 100mV supply voltage deviation from 1.1V nominal and across temperature from 0°C to 85°C. Figs. 3.7 and 3.8 show the self-Hamming distance as a function of voltage and sense amplifier read margin across temperature. The demonstrated 0 self-Hamming distance across $V_{DD}$ variation is an important measure of the system and circuit's robustness to supply noise and $V_{DD}$ droop in worst-case conditions. The demonstrated 0 self-Hamming distance across temperature variation (at all temperature extremities) demonstrates the system and circuit's robustness to environmental variation and shows the system is reliable across environmental operating conditions.

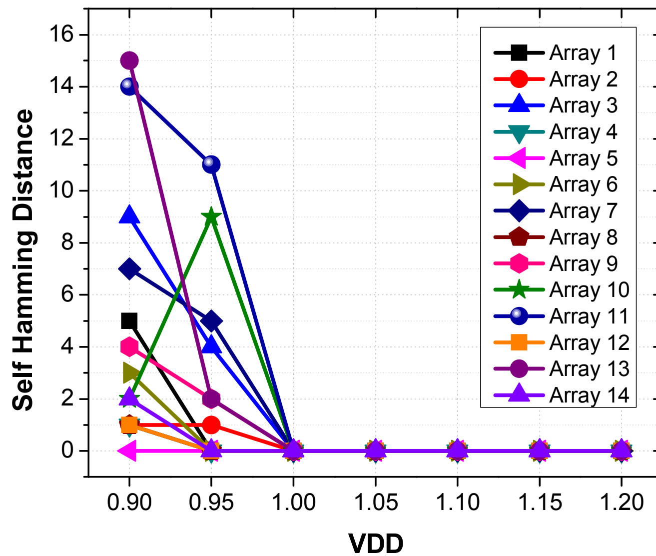



**Figure 3.7: Number of self-bit flips (robustness) vs $V_{DD}$ variation for 14 arrays.**

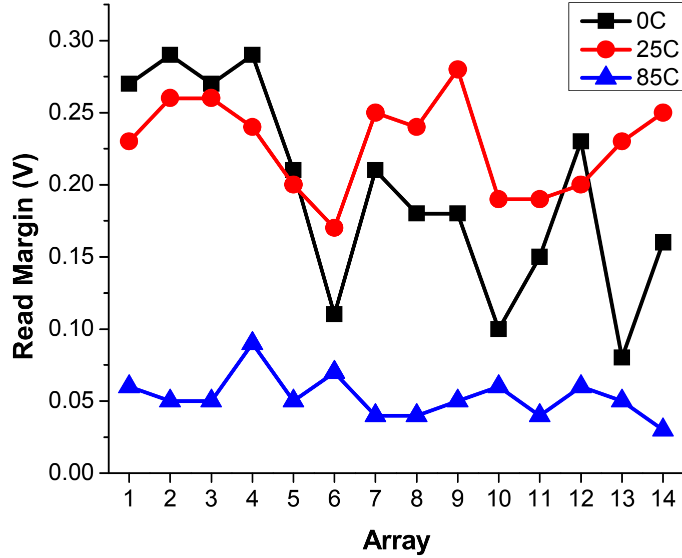
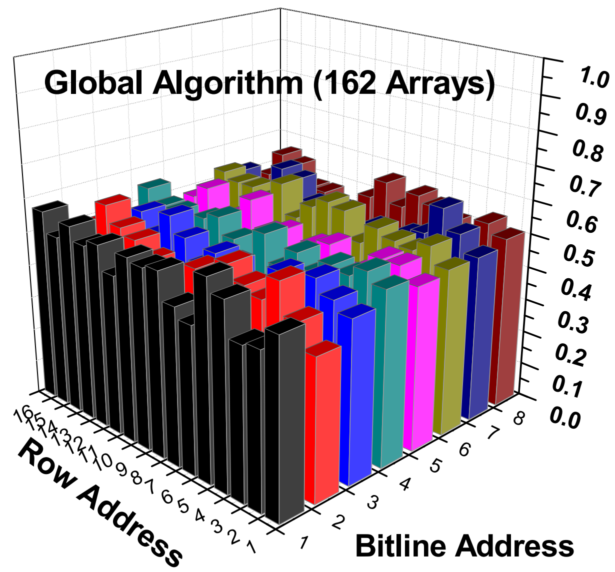



**Figure 3.8: Sense amplifier read margin (robustness) over Temperature for 14 arrays.**

Figure 3.9 shows the generated bits for each cell location, averaged across all arrays, with no obvious spatial artifacts. The top portion is for the global algorithm, while the bottom portion of Figure 3.9 is for the canary algorithm. As expected for the global algorithm, due to the nature of ensuring an equal number of one's and zero's in the algorithm, the average output for all locations is roughly around 0.5. Accordingly, for the canary algorithm, since it uses a canary row to determine stress cessation, the spread is larger (as shown in the Hamming distance Figure 3.5) and the average output for cells is less than 0.5; the average outputs are roughly around 0.4.



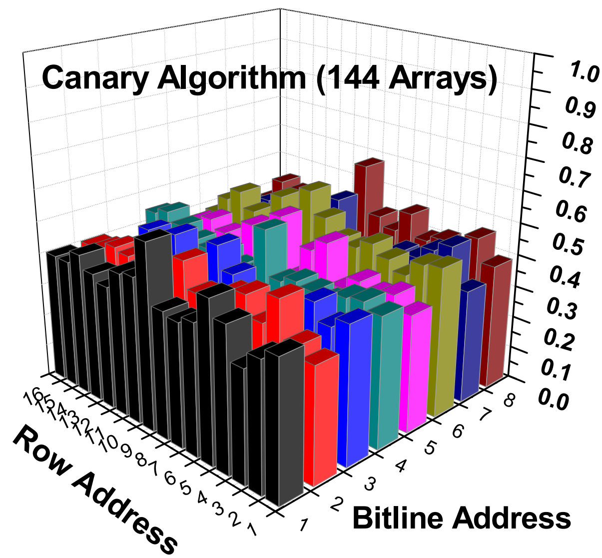




**Figure 3.9: Spatial distribution of bit probabilities for global algorithm (top) and canary algorithm (bottom).**

The spatial distribution of the breakdown time of each oxide in a typical array is shown in Figure 3.10. This example spatially demonstrates the distribution in when specific cells in an ID array begin to read non-zero read margins using the sense amp, demonstrating when they have begun to have become broken during the quick stress iterations of the algorithms. As seen, the spread across times is large, and there are no obvious spatial concentrations of cells with the same burn times, showing that the circuit design has no obvious spatial biases (ie such as due to stress power supply distribution). The average distribution histograms will be shown later (below) in Figure 3.11.

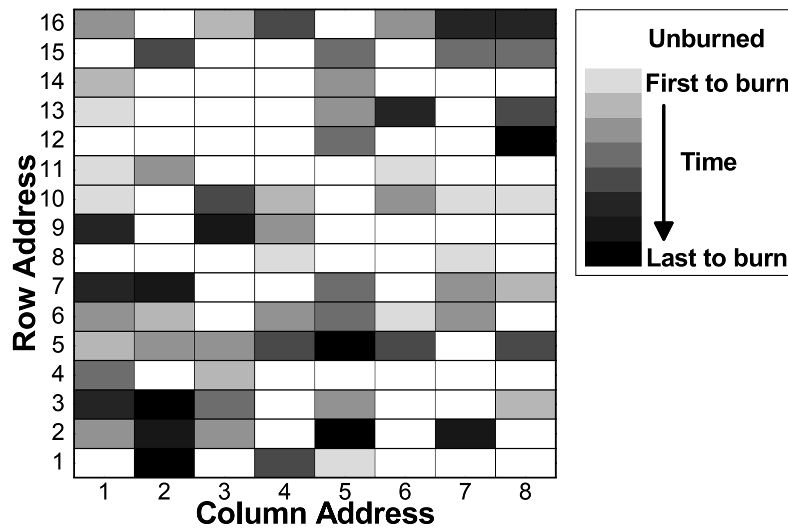



**Figure 3.10: Spatial distribution of the breakdown time for a typical oxide.**

The read power is 0.34 pJ per bit (Table 3.1). Table 3.1 provides a comparison of OxID to related prior work [17,18], showing improved energy, stability, and density. Note that the average unstable bit for [17] was not available, so there cannot be a comparison with that work for that metric. However, [17] relies on Vth mismatch which

is inherently prone to small average unstable bit readings similar to those mentioned in [18]. Since the technology for [17] is much slower, the lower throughput is to be expected; however, for [18], it is unclear why the quoted throughput is not higher, since it is essentially a custom SRAM array. As seen, the area is roughly comparable to that in [17] after technology scaling, and roughly 3X better than [18] after technology scaling.

| Ref. | Through-put (Bps) | Energy per bit (pJ/bit) | Average Unstable Bit | ID Length | Tech. (nm) | Area ($um^2$) | Area Scaled to 65 nm ($um^2$) |
|---|---|---|---|---|---|---|---|
| Lofstrom et al [17] | 3.75k | 8330 | N/A | 112 | 350 | 23,496 | 939.84 |
| Su et al [18] | 125k | 0.93 | 3.9 | 128 | 130 | 15,288 | 3822 |
| This work | 625M | 0.34 | 0 | 128 | 65 | 1242 | 1242 |

**Table 3.1: Comparison with previous works.**

Figure 3.11 shows the number of stress intervals across all arrays. As mentioned above in the discussion regarding the spatial distribution of burn times for a sample array, overall, for all arrays averaged, clearly the global algorithm has a tighter distribution as well as a higher mean. This difference is due to the inherent difference in the nature of the two algorithms; the global algorithm ensures an equal number of ones and zeros before cessation, while the canary algorithm stops immediately as the canary row reaches an equal number of ones and zeros.

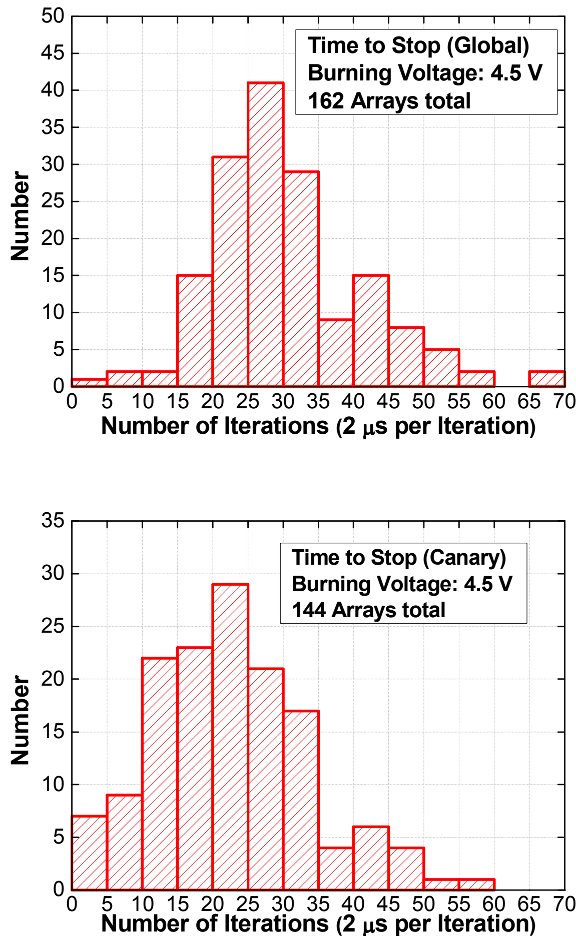



**Figure 3.11: Stress intervals required for global (top) and canary (bottom) algorithms.**

Figure 3.12 shows the chip microphotograph and chip statistics are included in Table 3.2. The two cores are vertically tiled and the scan chains are interconnected. There are 8 array banks (ID generators) per core, totaling 16 ID generators per chip. Each core has its own separate controller and clock generator. Probe pads are placed along the periphery and used to test all nodes of a discrete 3-T cell, for specific verification of on-chip performance.

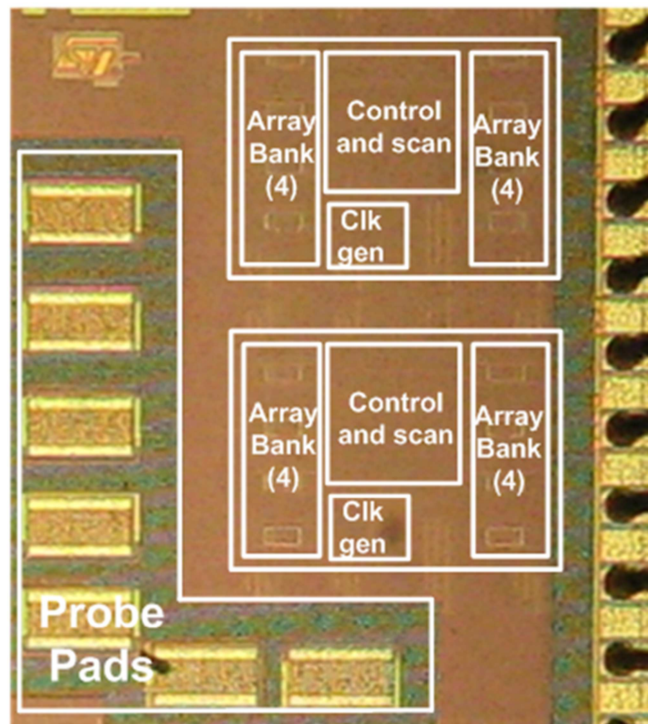



**Figure 3.12: Die photo.**

| | Global Algorithm |
|---|---|
| VDD | 1.1 V |
| Bit Length | 128 |
| Throughput (bps) | 625 MB |
| Average Hamming Distance (162 arrays) | 63.92 |

**Table 3.2: Chip summary.**

## 3.7 Summary and discussion

OxID presents a permanent chip ID that is randomly generated. The permanence identifies the chip indefinitely, improving security and safety for applications that require such a feature, such as many security and digital rights management applications. Another important benefit specific to OxID over previous on-chip ID systems is that the ID remains 0 until first use, when it ***must*** be generated; thus, prior security intrusion into the system to read the ID cannot be completed without the intruder generating the ID. But once the intruder generates the ID, the ID intrusion is by definition detected.

Further ways for shrinking down the area of the OxID cell have been explored, and remain to be implemented in hard silicon. In particular, the thick-oxide transistor takes up a lot of area in the layout due to DRC spacing rules regarding thick-oxides; by replacing this thick-oxide transistor with a series of thin oxide transistors (three or five, for example) that reduce the high voltage gradually using multiple diode drops, area savings by as much as half can be realized.

# CHAPTER IV

## OxiGen: A True Random Number Generator using Time-Dependent Dielectric Breakdown

This chapter explores how to further exploit variation optimally, using automated self-calibrating algorithms for tuning the on-chip circuitry. It discusses, in particular, the variation inherent in gate oxide breakdown and how to dynamically adjust for it to maximize the quantity of random bit generation output. We examine the process of putting a gate oxide repeatedly in soft breakdown, and then using the time to soft dielectric breakdown repeatedly to generate a binary count value on-chip. This system and method is called OxiGen. The chapter begins with a brief introduction to random number generation, in section 4.1. In section 4.2, we briefly survey hardware true random number generation and previously accomplished work on it. In section 4.3, the OxiGen system and method is described. In section 4.4, we discuss OxiGen's measured results. Finally, we summarize our findings in section 4.5.

In addition to demonstrating a new, more robust and stable way of generating truly random numbers, this chapter also illustrates another way to exploit variation to our

benefit. While previous techniques for true random number generation have exploited the randomness of the environment, such as thermal noise and telegraph noise, this technique serves as an example that device characteristics are a very viable resource to be taken advantage of. Because device characteristics will vary due to process variation, leveraging device characteristics also means adapting to and also leveraging process variation. As seen in this chapter, leveraging device characteristics and their associated variations can provide even better results than traditional noise sources.

## 4.1 Introduction to Random Number Generation

Random number generation plays a crucial role in cryptography and security. For example, public key cryptography systems demand strong key pair generation to ensure a third-party cannot decrypt secret messages. Traditionally, random bit sequences are generated in digital systems using pseudo-random number generators, which produce sequences that are not truly random but contain exploitable patterns, such as repetition and correlation. True random number generators (tRNGs) use physical phenomena as a randomness source. Before the advent of modern computing, random numbers were derived from numerical tables such as census reports, mathematical tables, and phone directories; by actual random events such as rolling die; or by using a calculation of some sort [24]. The former two were clearly related to true random number generation, while the latter was pseudo-random number generation.

There have been numerous implementations of hardware random number generators that have integrated the random number generator on-chip. These will be

briefly surveyed in section 4.2 First, however, it is worthwhile surveying the numerous methods that abound for pseudo random number generation (pRNG). A very common pRNG is the linear feedback shift register (LFSR); the LFSR is simply a shift register that cycles through a fixed number of states based on an input determined by its previous state. Because of its fixed number of states, a LFSR will eventually repeat itself and is thus predictable; hence it is pseudo-random. LFSR's have been implemented in hardware [25] and also software.

Another very common approach for pseudo random number generation is the linear congruential generator [26]: it takes a seed value $x_0$, a modulus *m*, a constant *c*, and a constant *a*, and produces the next bit in the random sequence via the equation: $x_n+1 = (a*x_n + c)$ modulus m. Because the periodicity of the random sequence depends on *m*, *m* is often taken to be as large as possible, such as $2^{32}-1$ on a 32-bit machine.

The multiply-with-carry (MWC) pRNG is also particularly interesting due to its simplicity yet good performance [27]. It is similar in concept to the linear congruential generator in that a previous $r^{th}$ bit ($x_{n-r}$) in the random sequence is multiplied by a constant *a*, and added with a number $c_n$. Instead of *c* being constant as with the linear congruential generator, *c* is now also a function of the current $n^{th}$ bit being generated. The sum is then taken modulus a constant *b*, equivalent to the constant *m* of the linear congruential generator. The first r values of $x_n$ are initialized with random seed values; in fact, $x_0$ up to $x_r$ are the r residues of *b*. The $c_n$ is calculated using the formula: $(a*x_{n-r} + c_{n-1}) / b$ .

Other advanced pRNGs exist, that are far beyond the scope or pertinence of this work to explain. The Mersenne twister [28], for example, developed in 1997, generates very high-quality pseudorandom numbers. Others exist such as the Lehmer random number generator [29], a variant of the linear congruential generator.

Pseudo random number generators suffer from periodicity, and thus have distinctly recognizable patterns that show up on spectral tests. Periodicity is a security drawback, and also generates biased and very questionable results when used in scientific simulations. Even with high periodicity, complex scientific simulations that require extremely high-order quantities of random numbers will still not be immune to periodicity.

In addition, even with high-quality pRNGs that have high periodicity, they can suffer from predictability once their parameters are known. This predictability is clearly a security vulnerability that can not only bias sensitive simulations but also prevent the pRNG from generating cryptographically secure random numbers. For example, although the Mersenne twister generates extremely high-quality pseudorandom numbers, its sequence can be predicted from knowing only 624 numbers (for its 32-bit word version)! To compound their weaknesses further, improperly chosen constants for these algorithms, due to insufficient analysis, can result in extremely weak and repetitive random numbers [39].

Thus, periodicity issues, software biases either purposely or inadvertently introduced in programming and algorithm design, seed values and constants being known, and predictability, all hamper the security and reliability of pRNG random

number outputs. Pseudo-random number generators, being algorithmically based, are also prone to cryptanalysis attacks [30]. Clearly, a hardware implementation that extracts a physical source of randomness is able to overcome all of these adverse effects. The difficulty with hardware implementations, however, lies with eliminating all sources of bias both during design phase, and device use; device use is a concern in the case of security, such as the user determining the source of randomness and deliberately injecting known biases to bias the output stream.

## 4.2 Hardware True Random Number Generation

Prior to the advent of modern computing, hardware number generators had existed. For example, in [24], the hardware exploited the random nature of radioactivity, by counting the number of output pulses of a G-M tube due to a radioactive element [24] and serializing the last few digits of the count to transform the normal distribution into a uniform distribution. More recently, in the past two decades, VLSI architectures using cellular automata models have been proposed to take advantage of parallelization to quickly generate random numbers [31 - 36]. Analysis has also been carried out on random number generators for parallel computing [37].

Even more recently, the importance of having a robust, reliable, and secure on-chip random number generator has grown. The ability to quickly generate random numbers in the CPU core itself, as opposed to on separate discrete chips that then pass the output through various hardware stages, results in increased security and orders-increased throughput, making faster scientific simulations, such as those based on Monte Carlo

methods, possible. For example, the Intel RNG [38] makes a clear case for the need for on-chip tRNGs as opposed to hardware or software pRNGs, and is based on "sampling thermal noise by amplifying the voltage across undriven resistors" [37]. While the resulting bitstream is demonstrated to pass statistical tests for true random numbers, the approach relies on amplifying extremely small noise, which requires complex and sensitive circuitry, and digital post-processing steps. Both of these weaknesses are elaborated upon below. Via, another major CPU maker, also has on-chip hardware RNGs included; rather than using thermal noise, Via's on-chip RNG was based on clock drift [40]. However, it also relies on hardware post-processing of results. Clearly, there is a pressing need for on-chip tRNGs, as demonstrated by major CPU vendors such as Intel and Via including their own on-chip tRNGs.

We first briefly survey other previous on-chip tRNG architectures; afterwards, we will elaborate further upon them. The previous on-chip tRNG architectures have used telegraph noise [41] and thermal noise as the physical source [42-48]. Thermal noise is often used indirectly with a metastable inverter [42-45], a jitter-prone oscillator [46,47], or a discrete-time chaotic pipelined structure [48]. Another approach used fluctuating gate oxide current after soft breakdown (SBD) as a noise source [49]. However, most prior architectures [41,42,44,47,48] rely on post-processing to remove bias in the generated stream, such as a "Von Neumann Corrector", which brings into doubt the randomness of the initial bit steam. Architectures that do not require a post-processor [43,45,46,49] have only passed 7 of 15 statistical randomness test in the NIST 800-22 benchmark [50], the accepted standard test for true randomness.

To elaborate further, the work of [43] is a tRNG based upon amplifying thermal noise. This work had the benefit of not requiring a post-processor, and applied statistical calibration towards maximizing the entropy of the random source being captured. It applied the standard method for harvesting thermal noise in a digital fashion: a latch that is put into metastability and which falls towards 0 or 1 based upon thermal noise. The key advantage of such latch based techniques is that it is quite digital; the positive feedback sidesteps the need for complex, noise-sensitive analog circuitry. The key innovation introduced in this work was a time-to-digital converter used to grade the quality of the metastability event. It was able to pass 7 of the 15 NIST 800-22 tests.

In the work presented in [45], entropy is harvested "from differential thermal-noise at the diffusion nodes of a cross-coupled inverter pair to resolve out of meta-stability." In other words, thermal noise is amplified in a metastable latch, with a throughput of 1 bit/cycle. The innovation in this work is that it is an all-digital design, with metastability event quality grading and a digital self-calibrating finite state machine feedback to maximize randomness; thus complex analog circuitry that is sensitive to PVT and noise is avoided. They also demonstrate ability to voltage scale supply VDD down to 280 mV while maintaing operation. The work does not require a post-processor; it passes 7 of the 15 NIST 800-22 suite of randomness tests.

Similarly, another previous on-chip tRNG based upon amplifying thermal noise using a metastable latch was shown in [42]. The novelty stressed in this work is that a "DC-nulling RNG uses a running estimate of the median total noise as the prediction"; the prediction referred to is a predictive signal applied to the evaluating metastable latch,

allowing the removal of the sampled and predictable non-random sources: supply noise, 1/f noise, interference, and device mismatch.

The work of [42] also applies this method of using a predictive signal to eliminate known non-random biases in the random-capture circuit to another type of random-capture circuit. This second circuit is FIR-based RNG; the RNG "uses a linear-prediciton filter to remove correlated components of the noise before sampling." In essence, the filter takes the stored previous bits, examines the autocorrelation of these bits, and applies the appropriate prediction signal as a voltage into the sampling circuit; this voltage will "remove correlated noise components." This system uses a post-processor to increase the randomness of the results.

Jitter-prone oscillators such as that employed in Via's CPUs, described above, have also been presented in [46,47]. In [46], the fast oscillator is deliberately jitter-enhanced by having an amplified thermal noise source, increasing the statistical quality of the randomness of the bitstream output. The reported results for the work in [46] are not post-processed, but it has been designed with a post-processor option in its hardware to increase randomness of the output bitstream.

There have been works that have combined several different methods of random noise sources to produce the random output. In [47], a combination of amplified thermal noise, jitter-prone oscillator sampling, and a discrete-time chaos system is used. The precise RNG system amplifies the thermal noise, which is then summed into a A/D-based chaotic system. The chaotic system output then drives a CCO, which is then subsequently

sampled at a user-defined frequency that is lower. This sampling is accomplished using a typical D flip-flop. The system uses a post-processor.

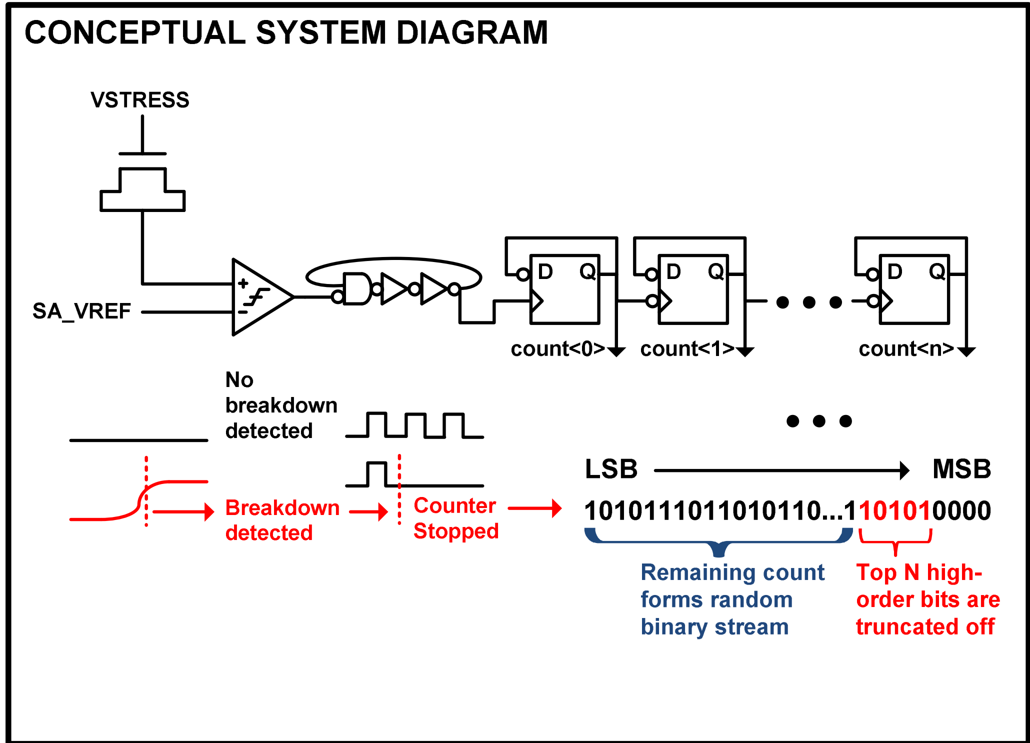
## 4.3 OxiGen Operation

We present a novel tRNG architecture based on a random physical phenomenon not previously used for on-chip random number generation. The architecture, called OxiGen, repeatedly forces soft gate oxide dielectric breakdown under voltage stress and uses the time to breakdown to generate a random bit sequence. This time to failure (TTF) is on the order of milliseconds, which is large compared to circuit speed. Hence, a key advantage of the approach is that TTF can be easily and accurately captured using a simple counter. This avoids the complex methods needed to capture thermal noise (the most common source of randomness), which is on the order of μV [43] and requires sensitive amplifiers.

The OxiGen method was implemented on an array of 128 MOS capacitors and can generate an estimated >5 billion true random bits before permanent breakdown of the capacitor oxides. OxiGen is the first tRNG to pass all 15 NIST tests without a post-processor. OxiGen was fabricated in 65nm CMOS, consumes 2 mW of power and 0.0012 $mm^2$ of area, and produces 11 kb/s of random data.

Breakdown occurs when a stress voltage is applied across an oxide, and is an inherently random process: for two completely identical oxides under the same stress conditions, TTF is different and unpredictable [51]. The proposed architecture measures

oxide TTF using a binary counter and generates a bitstream based on the resulting value; this is illustrated conceptually in Figure 4.1, below.



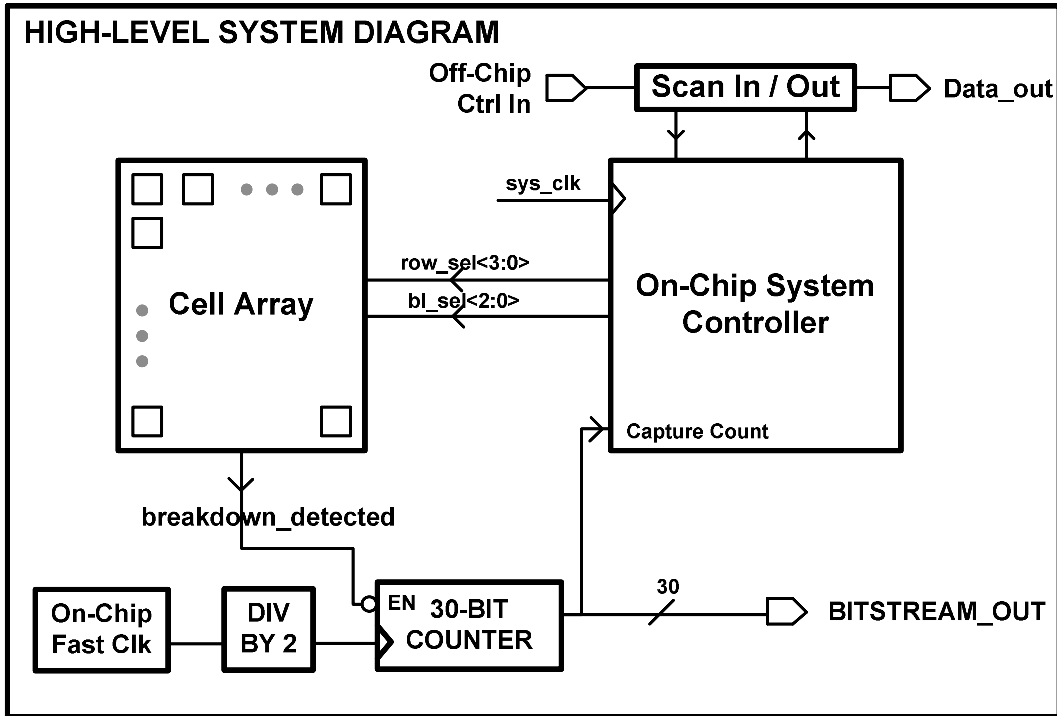




**Figure 4.1: Conceptual system diagram.**

Since TTF is normally distributed with finite variance, the counter value will have similar characteristics, whereas a generated random bitstream requires a uniform distribution of bits. The method resolves this by truncating the counter value, discarding n high order bits while keeping remaining lower order bits and outputting them serially to create a random bitstream. First, the high bit of the binary counter value is searched for, starting from the most-significant bit position. This most significant '1' bit is discarded, along with n-1 successive higher-order bits. The remaining lower-order bits are guaranteed to have rolled over at least 2n-1 times. Thus the algorithm intelligently truncates bits to maximize the number of bits generated while guaranteeing a high-quality random bitstream. This also adjusts for shifts in the mean count value due to variation in oxide thickness and wear-out conditions. Unlike post-processing, the algorithm does not observe or manipulate bits placed in the random bitstream, instead it judges the quality of bits by their position in the counter.

OxiGen relies on the fact that oxide breakdown occurs in stages. By carefully monitoring the bitline voltage, millions of random bits can be generated from each device by repeated stressing until the oxide fully breaks down. The array of 16 wordlines and 8 bitlines is made up of 128 basic 3-T cells [8,52] constructed from a MOS capacitor under stress, a thick-oxide high-voltage protection transistor, and access transistor (Figure 4.2 below). Each cell is stressed in turn while the bitline is weakly held by a keeper, and the bitline voltage is monitored by a simple comparator. During stress, a 30-bit counter runs at 325 MHz. As the oxide resistance drops, the comparator detects a voltage rise on the bitline and stops the high-frequency counter and disables the cell wordline to discontinue oxide stress. The counter value is read-out, truncated as described, and output as the

random bitstream. A stress voltage is applied to the next cell in the array and the process repeats. Total bit generation can be increased by adding more cells to the array, and bitrate can be increased by adding more comparators and counters for parallel count generation. This implementation shares one counter for 8 bitlines, while 8 counters would yield 8× higher throughput.

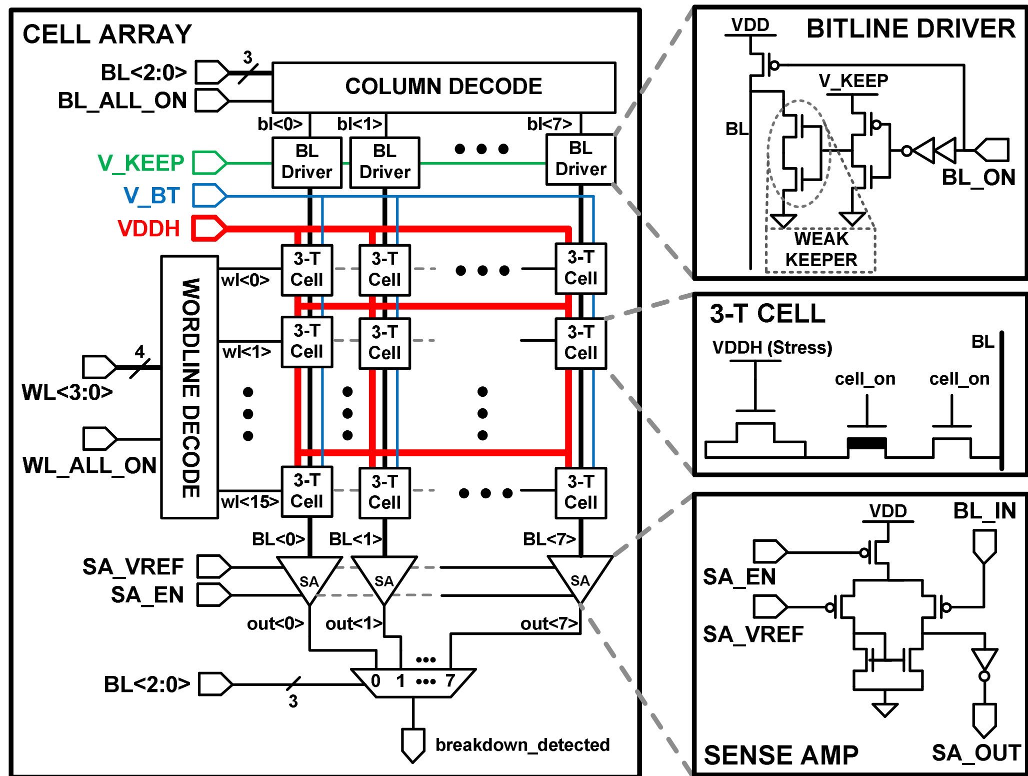



**Figure 4.2: Circuit diagram of cell array, showing 3-T cells, sense amps, and drivers.**

To maximize the total number of random bits generated by an array, OxiGen uses an algorithm to dynamically tune the stress voltage vddh and comparator reference voltage sa_vref. As cells in the array are cycled through and stressed, TTF is evaluated and sa_vref is adjusted accordingly. If TTF is too short, the reference voltage is increased

to generate longer counts, resulting in more random bits. Conversely, it is reduced if the reference voltage is found to be too high, causing the cell to be over-stressed without generating many additional random bits, wasting useful device lifetime. When sa_vref reaches it maximum value, the stress voltage vddh is reduced and sa_vref is reset to its minimum value. Thus, the algorithm dynamically converges on optimal conditions to generate the largest number of random bits, ensuring maximum cell harvest.

## 4.4 Measured Results

Two sets of random bit sequences generated by OxiGen were analyzed by the NIST 800-22 test suite for randomness. First, 300 sequences of 43k bits (13M bits total) were analyzed with different truncation lengths for 10 of the 15 tests. Figure 4.3 shows pass rates for NIST tests as a function of bits truncated, along with the minimum pass rate to be considered random and π‘s pass rate. When at least one bit is truncated, OxiGen easily exceeds the minimum pass rate and is comparable to π’s pass rate. The remaining 5 NIST tests require a larger sequence length and were tested with 100 sequences of 1M bits each (100M bits total); see Figure 4.4. OxiGen exceeded the minimum pass rate for the large sequence length tests and overall passed all 15 NIST tests with statistical significance and without post-processing.

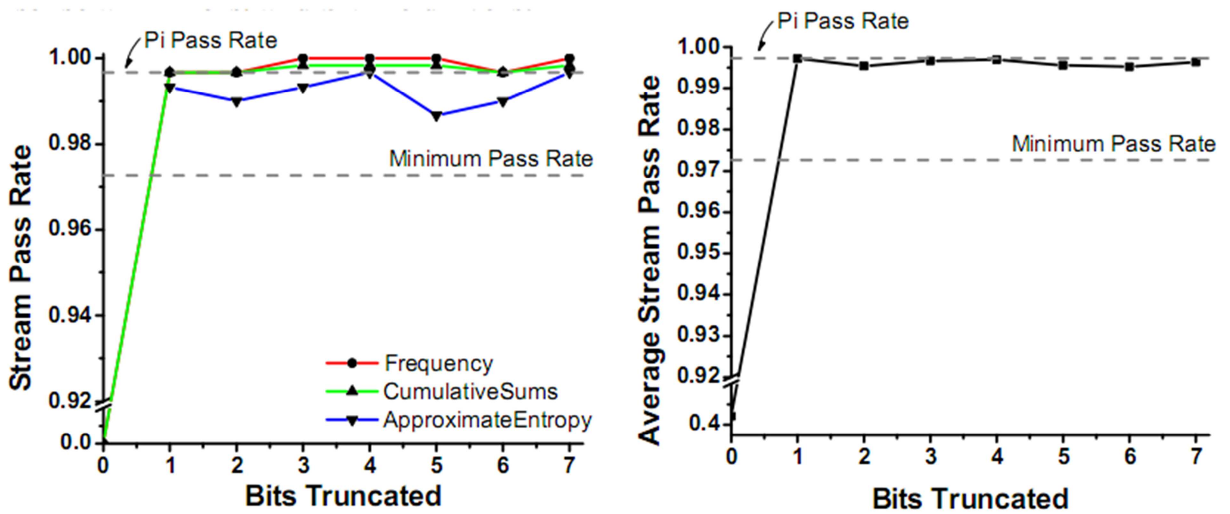



**Figure 4.3: NIST pass rates as a function of bits truncated.**

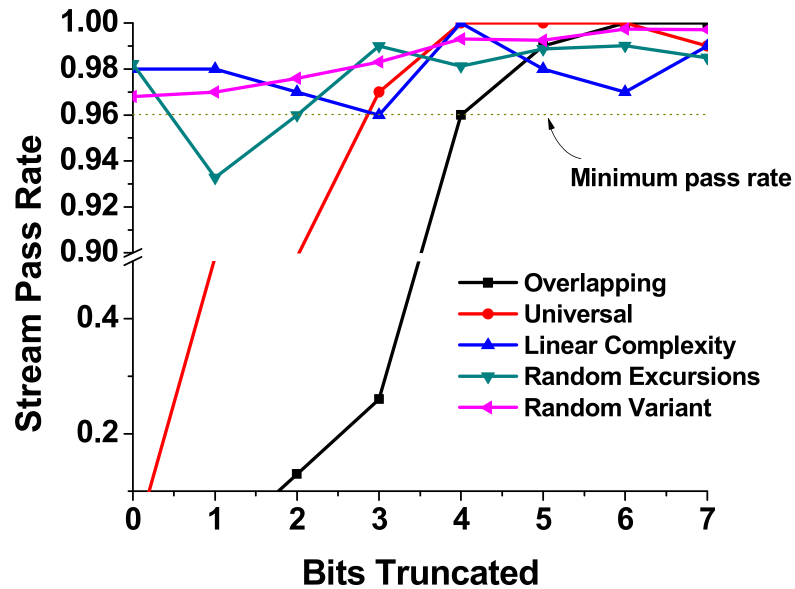



**Figure 4.4: Stream pass rates for NIST-required large sequence lengths.**

A visual plot of OxiGen's tRNG output compared to π and a traditional PRNG is shown in Figure 4.4. As seen visually, OxiGen's output after bits are truncated shows a random pattern. Figure 4.4 shows spectral tests [49] of a pseudorandom binary sequence

generated from a 7-bit LFSR, digits of $\pi$, and values from OxiGen before and after truncation. Spectral tests are scatter plots of random value pairs used to visualize autocorrelation of a bitstream. The LFSR shows autocorrelation while $\pi$ is random. OxiGen shows a random pattern after truncation.

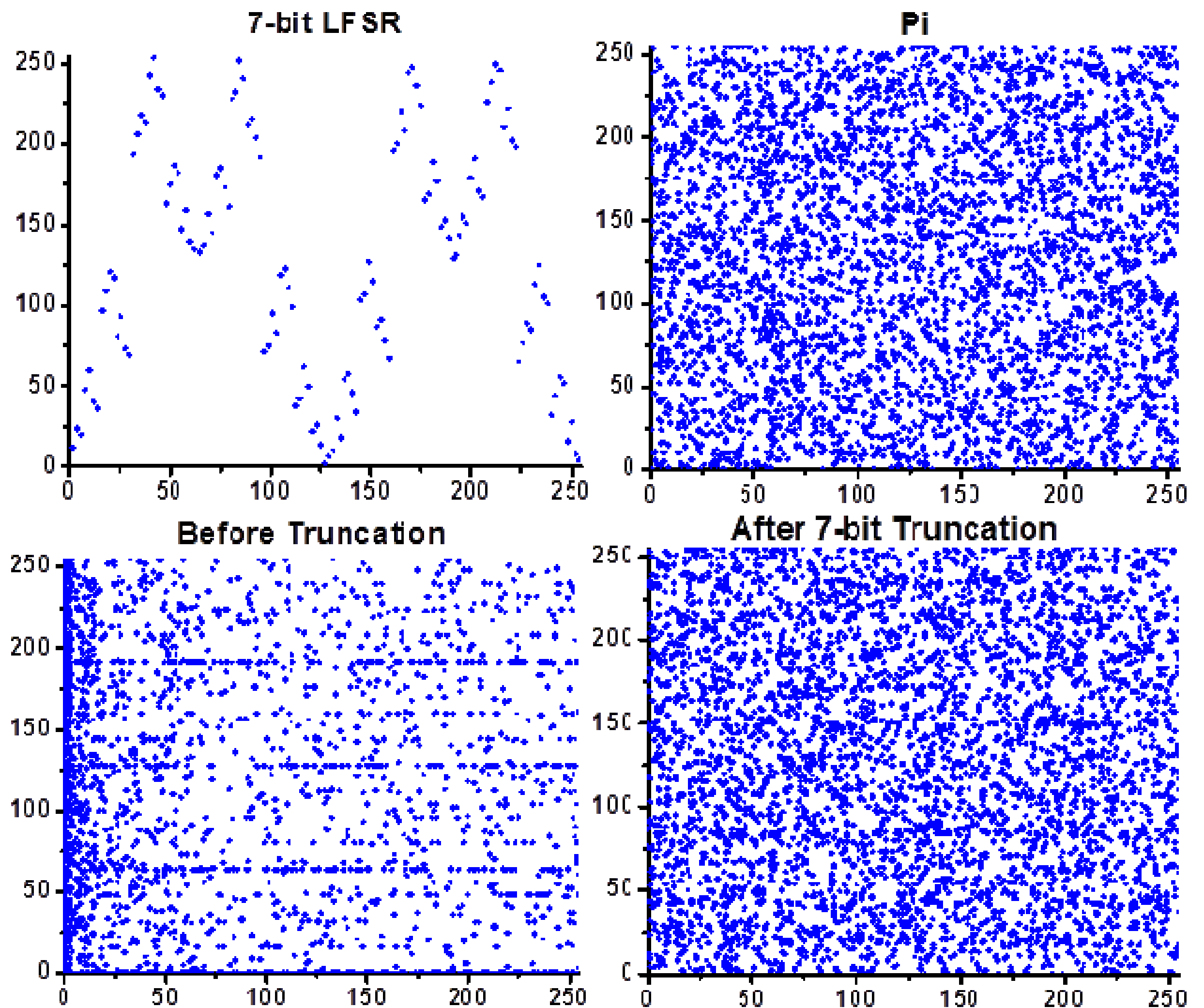



**Figure 4.5: Spectral tests [49] of a pseudorandom binary sequence generated from a 7-bit LFSR, digits of $\pi$, and values from OxiGen before and after truncation.**

In addition to spectral tests, a matrix plot of the data in black and white pixel format also visually demonstrates randomness, or lack thereof. Figure 4.5 below is such a matrix plot of 300 kb of data output of OxiGen, and shown for comparison are the matrix plots for the pseudorandom 7-bit LFSR and $\pi$ . The LFSR and OxiGen sequence before truncation demonstrate visible patterns, while digits of $\pi$ and the OxiGen sequence after truncation appear random.

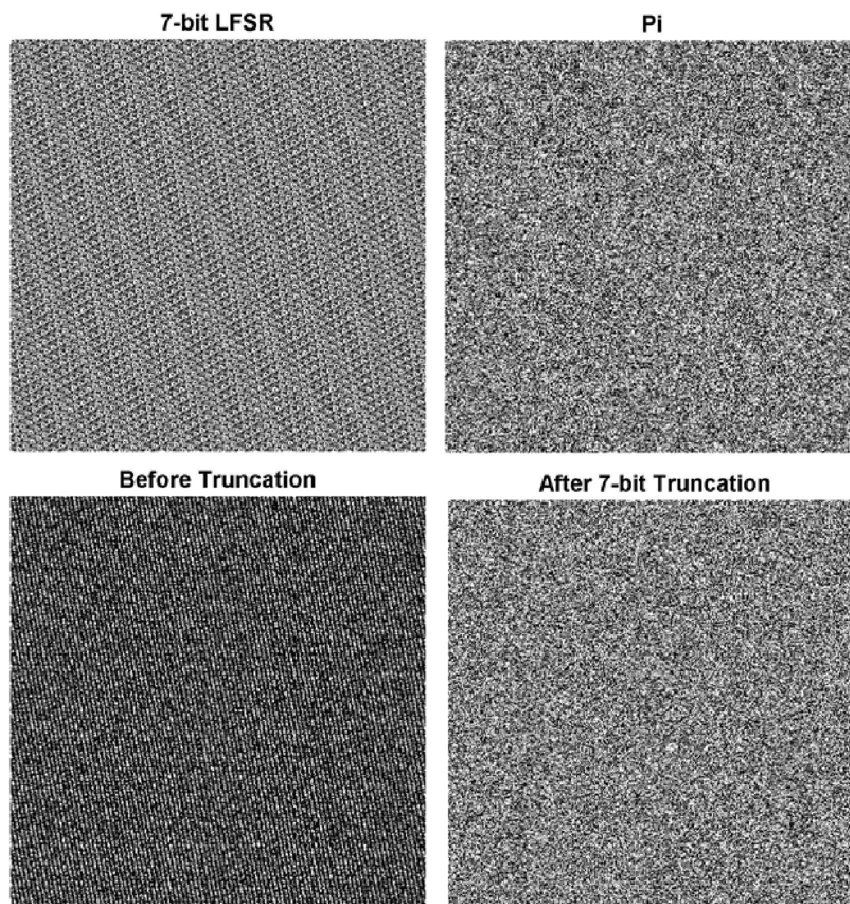



**Figure 4.6: Matrix of 300 kb each for pseudorandom binary sequence generated from a 7-bit LFSR, digits of $\pi$, and values from OxiGen before and after truncation. Bits were placed consecutively from top to bottom then left to right.**

Figure 4.6 below shows automated stress and reference voltage tuning behavior as observed while generating 500M bits over a 3 month period. The array was not exhausted at the end of the 3 month period. The comparator reference voltage had incremented less than half of its range and the stress voltage was still at the initial value of 3.5V across all cells, indicating the array was still in the early stages of life after generating 500M bits.

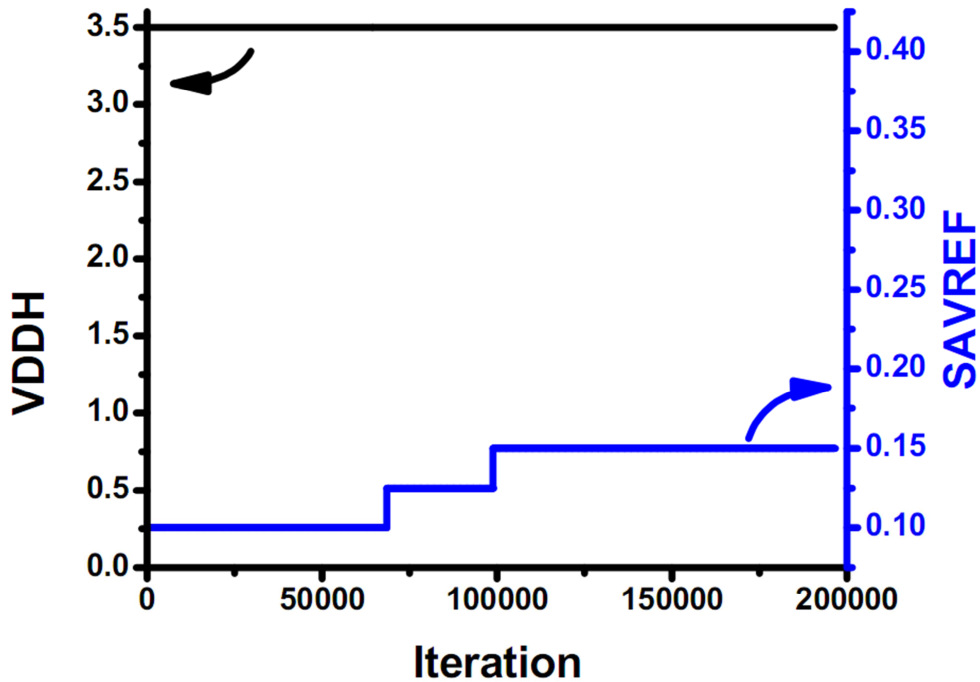



**Figure 4.7: Sa_vref and vddh for single cell during generation of 500Mb data. A typical iteration yields 20 bits.**

A typical internet secure session link (SSL) key is 256 bits. Assuming a new secure internet session every 5 minutes for 24hrs/day, 500M bits provides 10 years of operation, exceeding the lifetime of most mobile devices even if the bitstream is aggressively truncated during generation.

Furthermore, an accelerated test was performed on five cells, where the oxide was stressed 1000× longer than necessary, to predict the total number of bits that can be harvested from the cell array. Figure 4.7 shows the algorithm self-adjusting after a low bit harvest and then reconverging on the next optimal stress/reference voltage pair for maximum bit harvesting. Based on the progression of voltage tuning settings, the test showed at most only 5-10% of the array had been exhausted after 500M generated bits, indicating the array can generate >5B bits before permanent wear-out and giving an even greater safety factor compared to typical use.

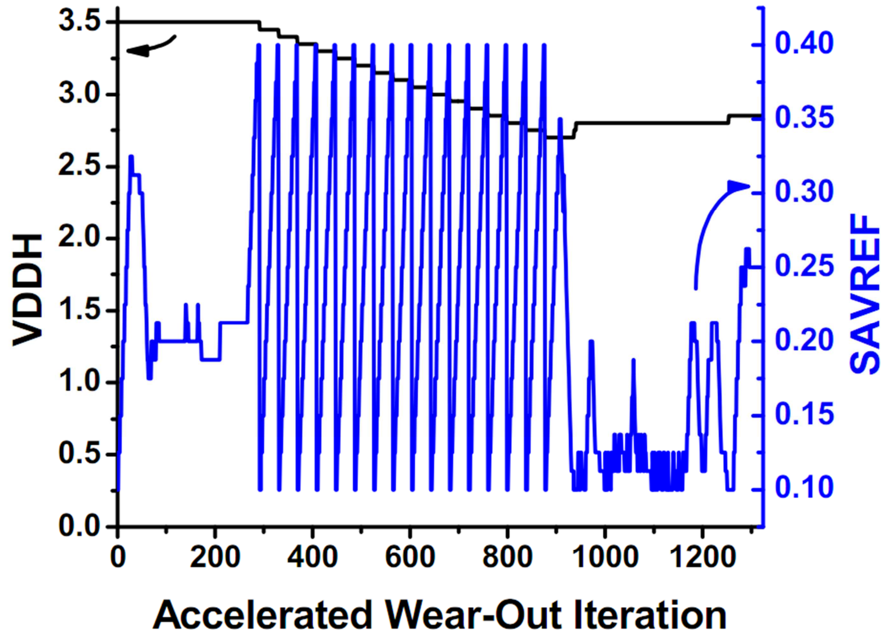



**Figure 4.8: Accelerated stress testing diagrams.**

Figure 4.8 below shows bit harvest for another accelerated wear-out run. This cell exhibits constant bit production and the algorithm does not need to adjust stress voltage. Note that the total bit harvest shown in the bottom graph is *per cell* (and data shown is for this cell).

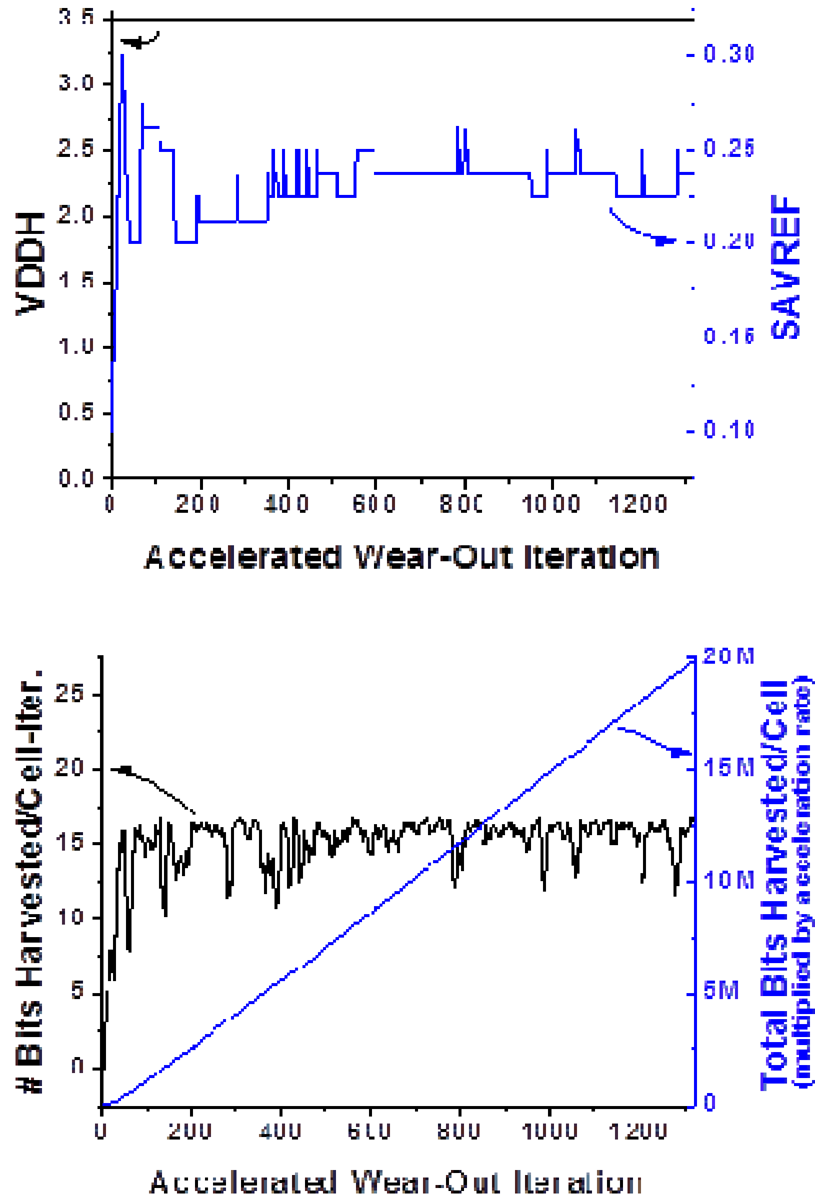



**Figure 4.9: This cell exhibits constant bit production and the algorithm does not need to adjust stress voltage.**

Table 4.1 below shows the detailed parameters input into the NIST 800-22 suite. For different stream lengths and numbers of sequences, the alpha value in the suite was adjusted for maximum statistical significance, following the instructions in the NIST 800-22 suite manual. Accordingly, note that all tests run were statistically significant, with much more than sufficient stream lengths, and sequence sample sizes (total bits).

| NIST 800-22 Test | Seq. Len. | Seq. Sample Size | Total Bits | Prop. of Streams Passed | Minimum Pass Rate | Pass Test | Statistically Significant |
|---|---|---|---|---|---|---|---|
| Frequency | 43 kb | 300 | 13 Mb | 1.0000 | 0.9727 | Yes | Yes |
| Block Frequency | 43 kb | 300 | 13 Mb | 0.9967 | 0.9727 | Yes | Yes |
| Runs | 43 kb | 300 | 13 Mb | 1.0000 | 0.9727 | Yes | Yes |
| Longest Run | 43 kb | 300 | 13 Mb | 0.9967 | 0.9727 | Yes | Yes |
| Rank | 43 kb | 300 | 13 Mb | 1.0000 | 0.9727 | Yes | Yes |
| DFT | 43 kb | 300 | 13 Mb | 0.9967 | 0.9727 | Yes | Yes |
| Non-Overlapping | 43 kb | 300 | 13 Mb | 0.9960 | 0.9727 | Yes | Yes |
| Overlapping | 1 Mb | 100 | 100 Mb | 0.9800 | 0.9602 | Yes | Yes |
| Universal | 1 Mb | 100 | 100 Mb | 0.9900 | 0.9602 | Yes | Yes |
| Linear Complexity | 1 Mb | 100 | 100 Mb | 0.9800 | 0.9602 | Yes | Yes |
| Serial | 43 kb | 300 | 13 Mb | 0.9984 | 0.9727 | Yes | Yes |
| Approximat e Entropy | 43 kb | 300 | 13 Mb | 0.9967 | 0.9727 | Yes | Yes |
| Cumulative Sums | 43 kb | 300 | 13 Mb | 0.9984 | 0.9727 | Yes | Yes |
| Random Excursions | 1 Mb | 100 | 100 Mb | 0.9903 | 0.9530 | Yes | Yes |
| Random Variant | 1 Mb | 100 | 100 Mb | 0.9923 | 0.9530 | Yes | Yes |

**Table 4.1: Table of all NIST 800-22 tests, sequence lengths, sequence sample size (number of sequences), proportions of sequences passed, and minimum passing rate, using random bits generated by OxiGen after 4-bit truncation.**

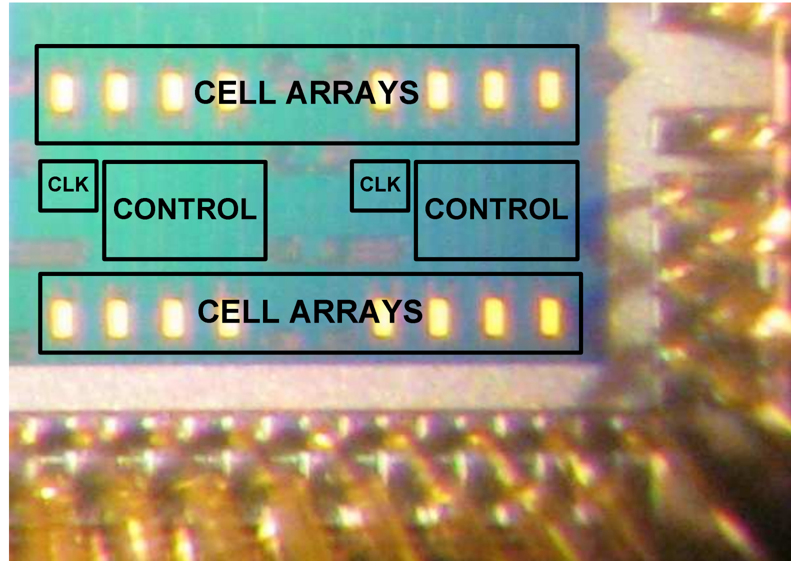



**Figure 4.10: Die photo of OxiGen.**

A die photo of the fabricated chip is shown above in Figure 4.9. Table 4.2 shows a summary of the chip technical specifications.

| Specification | OxiGen |
| --- | --- |
| Technology | 65 nm |
| VDD | 1.1 V |
| Clock Freq. | 650 MHz |
| Bitrate | 11 kb/s |
| NIST Tests Passed | 15 |
| Area | 0.0012 mm² |
| Power | 2 mW |
| Post-Processor | No |

**Table 4.2: Summary of OxiGen technical specifications.**

A comparison of OxiGen with prior tRNGs is given below in Table 4.3. In summary, OxiGen passes all NIST 800-22 randomness tests, which is a first for tRNGs not using a post-processor corrector.

| Ref. | Bitrate | NIST Passed | Tech. | Area ($mm^2$) | Area Norm. ($mm^2$) | Power (mW) | Post-Process |
|---|---|---|---|---|---|---|---|
| This work | 11 kb/s | All | 65 nm | 0.0012 | 0.0012 | 2 | No |
| [45] | 2.4 Gb/s | 7 | 45 nm | 0.004 | 0.0083 | 7 | No |
| [42] | 5 kb/s | * | 0.35 µm | 0.031 | 0.0011 | 0.0094 | Yes |
| [43] | 200 kb/s | 7 | 0.13 µm | 0.145 | 0.0363 | 1 | No |
| [41] | 50 kb/s | – | 0.12 µm | 0.009 | 0.0026 | 0.050 | Yes |
| [48] | 40 Mb/s | All | 0.35 µm | 0.52 | 0.0179 | 29 | Yes |
| [49] | 50 kb/s | – | Discrete | – | – | – | No |
| [46] | 10 Mb/s | 3 | 0.18 µm | 0.016 | 0.0021 | 2.3 | No |
| [44] | 2 Mb/s | – | 0.6 µm | – | – | – | Yes |
| [47] | 1 Mb/s | 4 | 2 µm | 1.5 | 0.0016 | 3.9 | Yes |

**Table 4.3: Table comparing OxiGen to prior true random number generators.**

## 4.5 Conclusion

OxiGen demonstrates that we can further exploit on-chip variation to our benefit. In this case, we have demonstrated that we can exploit the randomness inherent in gate oxide breakdown to extract random numbers that pass all 15 random number tests in the NIST 800-22 test suite. This method allows for easier capture of the random information: oxide soft breakdown is easily detectable with ten's to hundred's of millivolts of bitline swing, and time to breakdown is orders larger than a fast on-chip oscillator. In contrast, traditional methods rely on complicated circuits and on-chip control to extract random numbers out of thermal noise. A key feature, in addition, is that we are able to maximize the bit-generation output capability by applying dynamic run-time algorithms that optimally calibrate and tune the on-chip control and circuitry to minimally degrade oxide lifetime to maximize the total bit harvest.

# CHAPTER V

## Mitigating and Monitoring Variation in Subthreshold

We have explored timing variation using in-situ delay monitoring, via a pico-second accurate time to digital converter discussed in Chapter 2. This chapter explores and expands upon this same vector: monitoring variation via a self-timed, self-tuning transition detection circuit. We first briefly analyze the effectiveness of the general circuit techniques employed in the transition detection circuit, as applied to digital circuits in general. Then, we apply our proposed techniques to a specific case application: a subthreshold self-timed transition detection circuit. The transition detection circuit is designed using the aforementioned techniques that mitigate process variation in subthreshold operation. We analyze the effectiveness of the subthreshold PVT and PVT variation mitigation techniques, in the context of our case application of the transition detector.

## 5.1 Proposed Methods of Increasing Robustness to Variability in Subthreshold Circuit Design

When performance is not a crucial part of the correct functioning of a device, but power consumption is, operating the circuit in the subthreshold or near-threshold regions can yield power gains at the expense of performance. As $V_{DD}$ is scaled down there is a quadric decrease in dynamic energy, but more importantly for power, the current decreases exponentially with gate voltage in the subthreshold regime. Thus, for non performance-critical circuits, it is advantageous to operate the circuit in the near or sub-threshold regime.

However, in subthreshold circuit design, there are several key factors that make circuit design in this operational regime difficult, all centered around the exponential dependence of current to threshold voltage shifts: short-channel effects that can cause high variation in Vth; random dopant fluctuation (RDF) that can cause large random mismatch between FET Vth's; and P-N mismatch in strength as $V_{DD}$ scales below threshold voltage.

We propose several methods to increasing robustness to variability in subthreshold circuit design: keeping nfet and pfet stacks small (preferably less than two transistors), thus necessitating a self-precharging type logic family; asynchronous handshaking between successive stages to ensure that, even in the worst-case Vth corner where a successive gate performs too slowly, the successive gate still has the input for a guaranteed amount of time to correctly switch its output; sizing up W and L to reduce

mismatch resulting from RDF and length variation; and sizing up the critical transistor(s) in a stack (if a stack has to be used) to ensure that it can meet timing in all worst-case corners.

First, nfet and pfet stacks should be kept as small as possible, preferably a single-transistor stack. From simulation results across several different technologies with random mismatch, with even with 2 stacks, across Monte Carlo, a large increase in failures becomes evident. The reason for this is twofold: first, body effect can occur during certain input states, where a non-zero $V_{BS}$ increases the transistor Vth; second, since the transistor strength is now exponentially dependent upon Vth in the sub-Vth regime, a particular case where both transistors in the stack happen to have higher-Vth can cause the stack resistance to increase dramatically compared to its TT counterpart. However, it is not feasible in a real design to collapse even minimal 2-input gates down to 1; thus, we propose that at a minimum, 2-input gates be used but with their stacked gates replaced by self-precharging headers (or headers precharged through other means such as a clock or timing feedback from other gates). In fact, using this methodology, we can replace n-input gates by single-transistor stacked gates; in general, this approach will necessitate collapsing of the logic tree down into a wider, broader tree, which increases logic density but vastly reduces accumulated Vth mismatch in the worst case. This likely requires upsizing of gates in order to drive the increased fanout, which is actually a benefit in sub-Vth as gate capacitance plays a lesser role due to vastly reduced driver strengths, and that upsizing decreases short-channel and RDF effects on Vth variation (discussed later).

Second, there may be special cases in non-CMOS style custom logic where a gate generates output from an incoming signal, such as generating a pulse by feeding in an input slope through an inverter chain (ie transition detection). In the subthreshold regime, generating pulses can be especially susceptible to corner case failures, because of either Vth mismatch causing worst-case pulse shaving, or successive stages in the inverter chain having cascading P-N mismatches. In these classes of circuits, we propose to introduce a gate-level asynchronous handshaking; the successive stage has a feedback signal that is generated only after it is done computing, and this feedback signal is fed back to the prior stage as a completion signal for the prior stage to reset. In this way, the previous stage will always hold its input long enough for the next stage to compute its output entirely.

Third, transistor width and length should be sized up larger when possible; the 3-sigma Vth variation caused by RDF is inversely proportional to the square root of the channel area: sigma-Vth ~ $K/(WxL)^{1/2}$ , where K is a constant equal to 4 mV x um [53]. At 65 nm and newer processes, Vth mismatch caused by RDF is similar to other sources of Vth mismatch, and will begin to become the prevailing source of Vth mismatch [53]. Thus, it is crucial in sub-Vth design to size up gate lengths and widths to decrease both RDF and short-channel effects.

Fourth, as mentioned above in the first point regarding transistor stacks, we note that a transistor stack can be critically underperforming in corner cases where both transistors have higher Vth. Conversely, however, we can take advantage of this if we know which inputs are arriving later and thus in the critical path. The gate transistors, to which these inputs feed, can be sized up even more than conventional sizing metrics dictate, as we know that capacitance plays a lesser role in sub-Vth circuit performance

due to the vastly increased driver resistance. These critical paths through a combinational logic stage can be analyzed using static timing analysis to determine the critical gates and gates' inputs, and thus a CAD approach can be taken to generate a specialized library of cells for sub-Vth APR.

## 5.2 Application of PVT Mitigation Design Techniques: 100% Self-Timed Transition Detection Circuit

As an application of the aforementioned principles for robust sub-Vth design, we apply these principles to the design of a transition detection circuit whose ultimate role is for measuring timing margins in a sub-Vth to super-Vth processing core. This transition detection circuit was implemented in silicon on a stand-alone core in chip presented in [7, 10].

The in-core timing margin system requires that we be able to monitor hundreds of input signals at once. If done conventionally through the traditional means of muxing and XOR-type transition detection circuits, the resulting logic depth would be deep and possibly incur huge accumulated amounts of mismatch in Vth (Figure 5.2). Following our discussed principles, we seek to collapse the gate depth to as small as possible in order to reduce accumulated Vth mismatch; this requires up-sizing gates, increasing gate inputs, and increasing gate fanout, but is actually a benefit as it also reduces short-channel and RDF effects on Vth mismatch. A regular traditional transition detection block is shown in Figure 5.2; observe that for 128 inputs, 7 stages of OR-gates in the OR-tree are required; this means a large accumulation of Vth mismatch, making differing paths very unequal in

delay and thus making the circuit unreliable and highly inaccurate for subthreshold design.

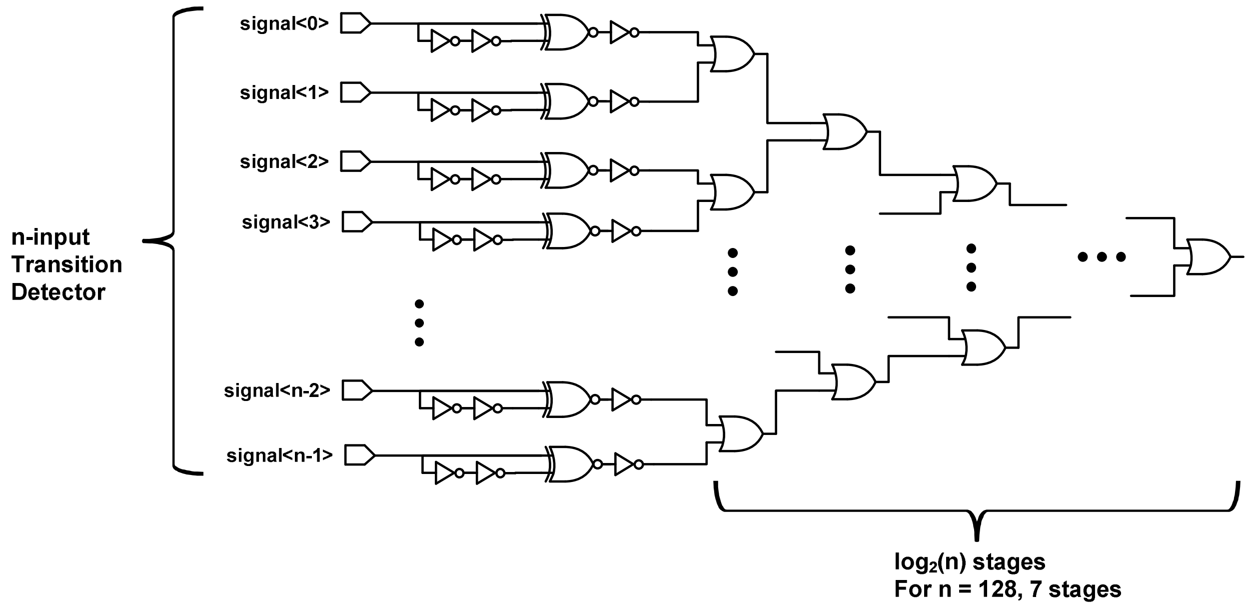



**Figure 5.1: A traditional implementation of a high-input CMOS transition detector with OR-tree**

Collapsing the logic depth necessitates a large increase in the number of inputs per custom gate. Therefore, it makes sense that we make the high number of input gates as compact as possible; that is, by keeping each input uniform in size and tightly bound together (fingered structure), we can increase matching and decrease process variation. Thus, for the transition detection circuit, it makes sense to use a wide-or structure where the inputs are all single nmos transistors and the pmos stack is replaced by a precharged header.

However, in a sub-Vth design, a precharged header that is clocked does not make much sense, as the variability between gates would mean the clock would have to be

severely frequency-margined in order for all gates to perform correctly. It makes more sense for the gate to be self-precharging; that is, it is weakly kept at precharge state by its header and a keeper, but as soon as its output node is switched, its output is fed back through an inverter chain and it is recharged after the inverter chain delay. The delay in the inverter chain determines the time to self-recharging after an input transitions. The wire-or structure is shown in Figure 5.2 below.

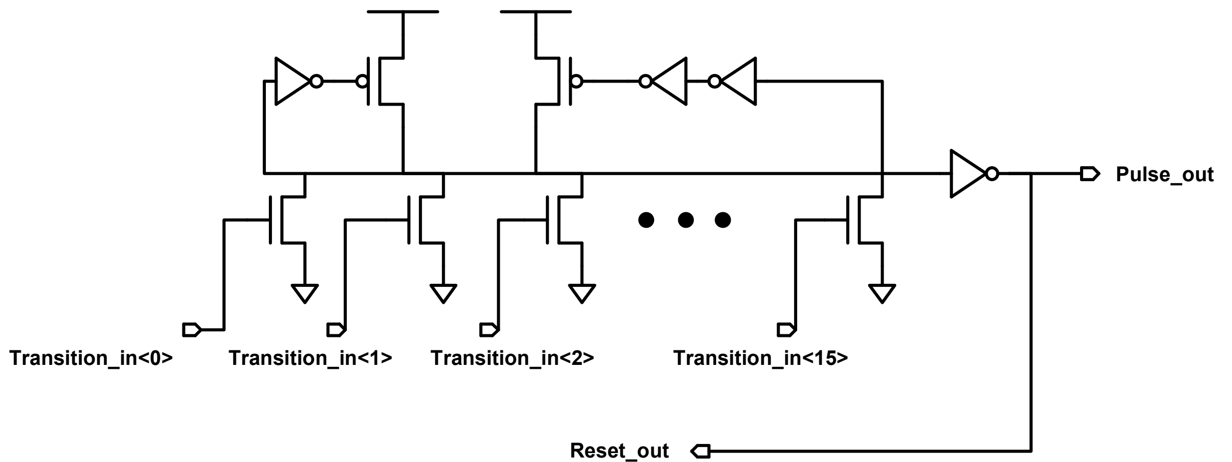



**Figure 5.2: Sub-Vth optimized self-timed wide-or structure for or'ing transition inputs.**

One constraining drawback of this approach is that as the number of inputs for the wide-gate increases, the distribution of arrival times at the inputs also increases, thus necessitating the self-precharging inverter delay to cover the worst-case of the spread of arrival times. We combat this necessity by noting that, essentially, two approaches can be taken to handling this constraint:

1) As noted, the self-precharging inverter delay can be lengthened to accommodate the worst-case arrival time. This causes the output pulse to

become, essentially, a worst-case pulse-width proportional to the time delay of the self-precharging inverter chain.

2) We size the self-precharging inverter delay to be short enough such that the output pulse meets the minimum pulse-width under all corners. We demonstrate that as successive inputs toggle, the pulse-width is actually linearly increasing and by measuring the final pulse-width, we can determine how many inputs were toggled.

The second approach offers the best advantages as long as the inverter chain is sized sufficiently for all corners; this condition is not hard to meet as long as the minimum output pulse-width required is not too short.

Below, we show a simulation result for the aforementioned second approach, which we had taken for the silicon implementation. In Figure 5.3, as a second input is successively delayed in time (x-axis), the output pulsewidth (y-axis) linearly increases by the arrival delay of the second transition, making it easy to interpret the arrival of a second signal and its arrival time by measuring the pulsewidth of the output pulse. This pulsewidth can be easily determined in an actual system if it is synchronized to a clock edge and therefore only the time of the falling edge of the pulsewidth needs to be captured (using a cheap TDC, for example). Note that the pulse width drops down to ~110 ns at t = 1.12 ns since the first transition has been reset at that point and the pulse width is due solely to the latter arriving second transition at that time.

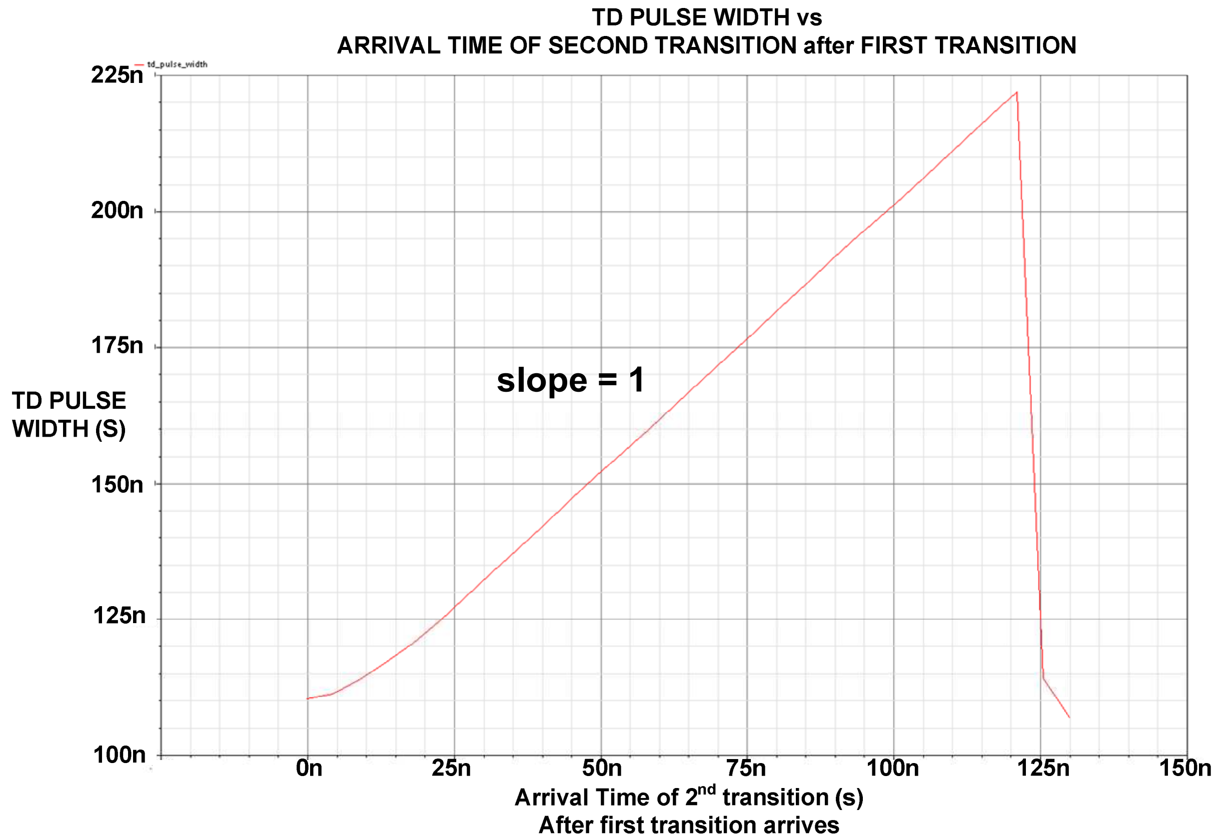



**Figure 5.3: Linear increase in TD output pulse-width due to second incoming transition**

In Figure 5.4, we show the TD pulsewidth output for separate instances of arrival times of the second transition as the second transition arrives 110 ns and more after the first transition. This is simply the output pulsewidth at t = 110n, 120n, 130n, 140n, and 150n, of the above Figure 5.3. The intent is to fully illustrate what is happening to the output pulsewidth in terms of its voltage waveform in time at later arrival times of the second transition, as it is not clear simply from Figure 5.3; that is, we are illustrating what happens around and after the 'drop' in Figure 5.3. Figure 5.4 shows that even for later arrival times of the second transition, we are still able to extrapolate exactly when the second transition arrived; after a certain arrival time ( > 120 ns), the TD output pulse separates into 2 separate pulses, which is also illustrated in Figure 5.4.

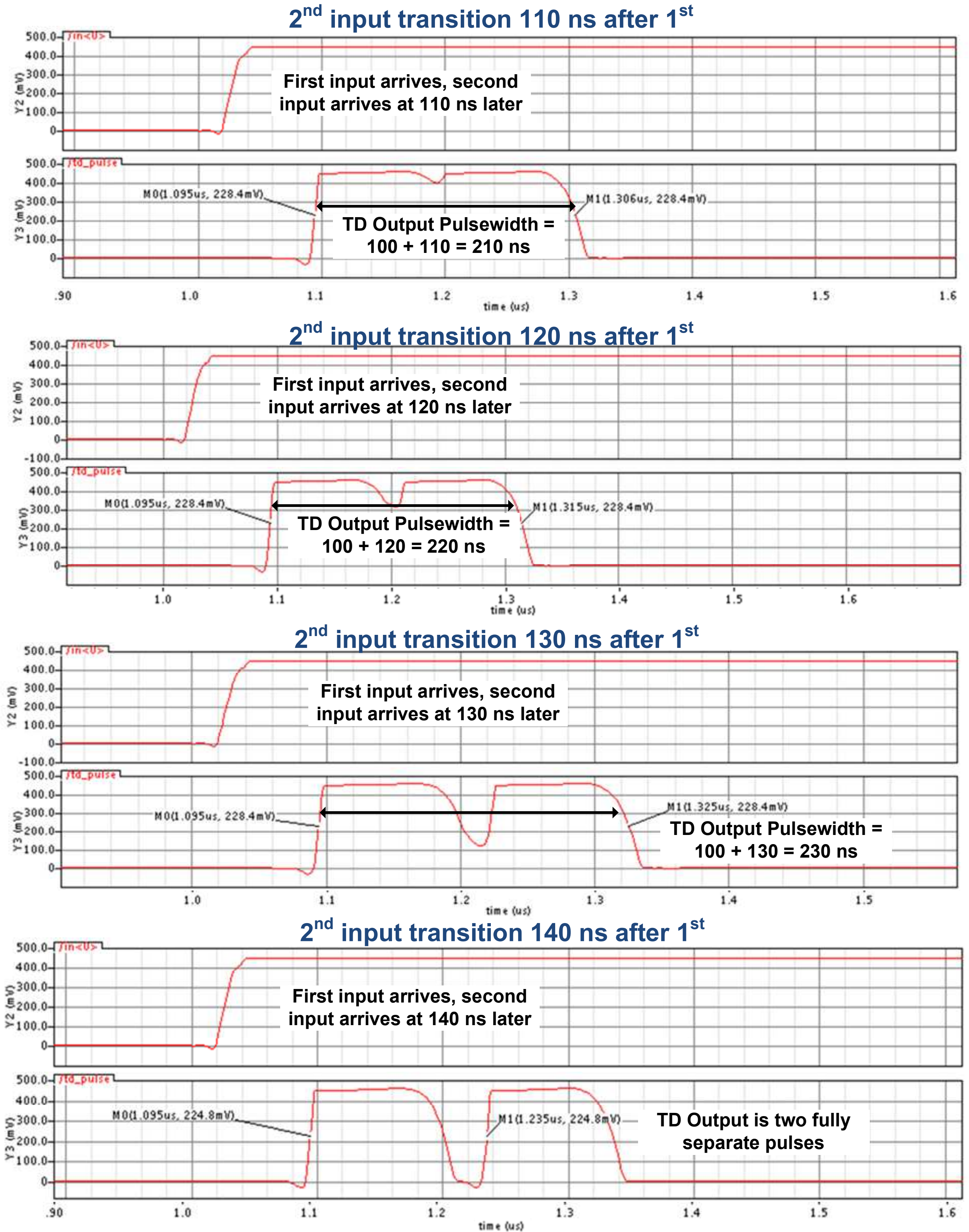



**Figure 5.4: TD Ouput Pulsewidth for later arrival times of 2nd Input Transition**

Next, we move onto the actual transition detection stage of the circuit. Since we want to keep the wide-or compact and tight in area, the transition detection circuits can be moved farther away to the locations where we actually want to detect the transitions. In sub-Vth, interconnect delay plays a much lesser role in signal timing [53,54] as driver resistance completely dominates over interconnect resistance and capacitance effects. Thus, it is not necessary that the transition detection stages be close in proximity to the wide-or stage.

The actual transition detection stage is shown below in Figure 5.5. The stage detects both rising and falling transitions, by feeding the monitored signal into an inverter chain. By tapping odd, opposing points of the inverter chain, we can feed these into a 2-input self-timed dynamic PMOS NAND gate to capture the generated pulse. We lock the pulse at the NAND gate and feed the output into the wide-or; the NAND gate is has a resetting footer that only resets once the wide-or has completed transitioning, and sends the reset signal back to this resetting footer. One constraint is that, as the wide-or has many inputs, if several transition detection stages switch in succession, the first ones would have to wait until the last one have finished in order to be reset. This means the first ones could possibly miss future transitions that toggle at its input. Thus, we add a second self-resetting footer in series with the feedback resetting footer; this footer waits a worst-case designed amount of time before it resets the NAND gate. Thus, this inverter delay imposes a design constraint that it must be sized to be long enough to cover the worst-case transition time of the wide-or. However, since the wide-or is a single-stack gate, the worst-case time is far better than equivalently sized CMOS gates (that would be

stacked and operating in sub-Vth with Vth variation penalties and stacked-gate effect on Vbs that adversely affects performance).

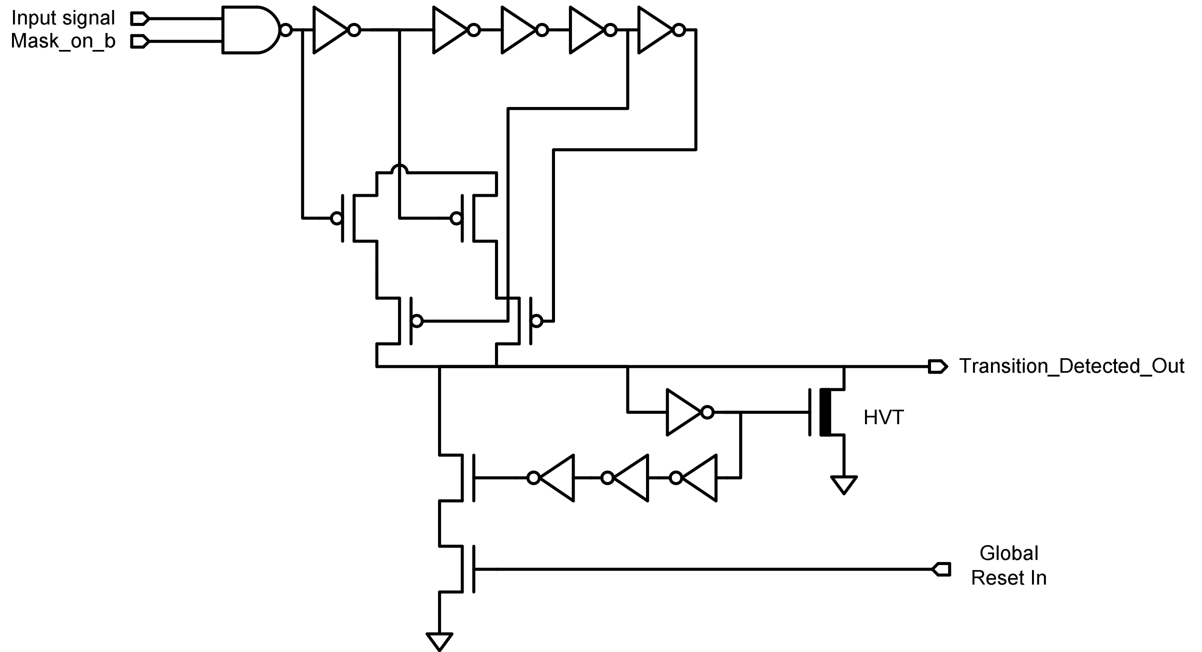



**Figure 5.5: Self-timed Transition Detection stage with asynchronous feedback input from wide-or stage.**

Finally, we show, below in Figure 5.6, the entire transition detector with the transition detection stages (16) connected to the wide-or stage (16 TD inputs). Note the self-timed aspects of each stage individually (TD: self-timed stack delay; Wire-or: self-timed pull-up recharge), and the self-timing between stages with the forward (TD_OUT) and reverse handshaking (pulse_out).

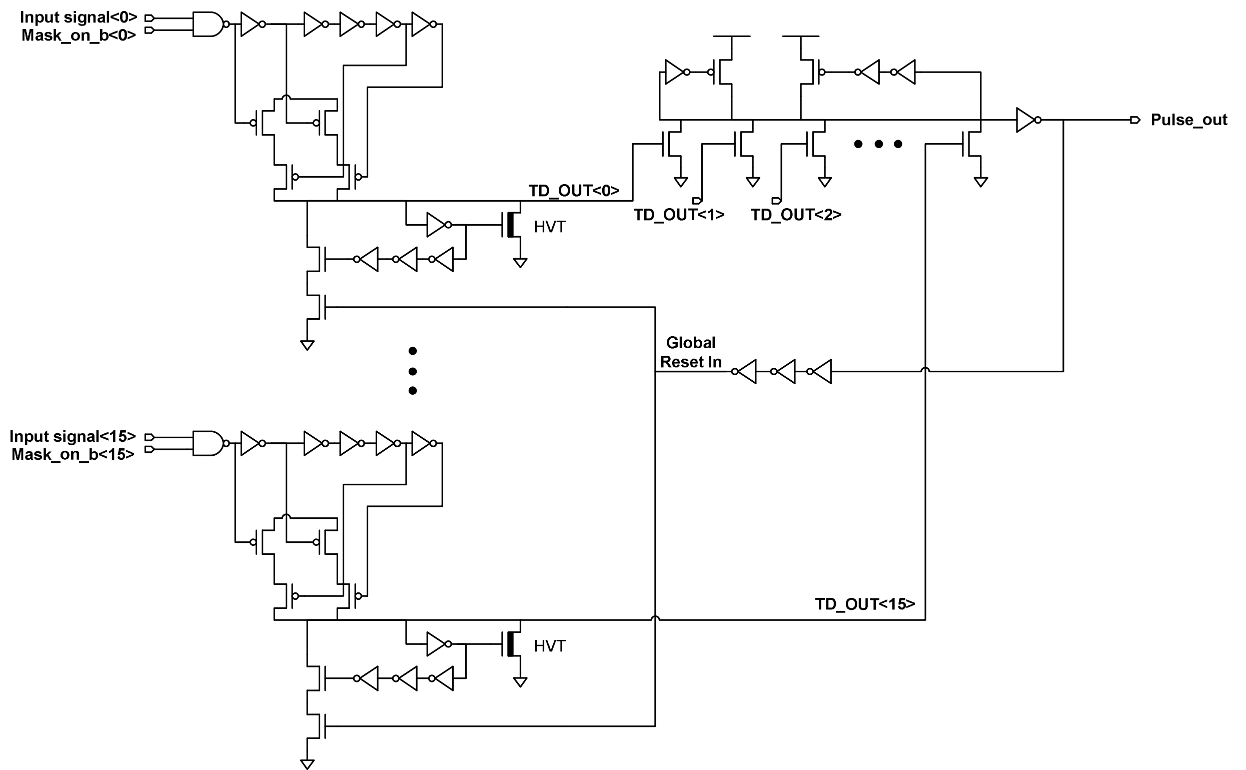



**Figure 5.6: Entire 100% Self-timed Transition Detector 100% Robust for Sub-Vth.**

# CHAPTER VI

## Conclusion

In this dissertation, we have discussed the important implications of variation as technology scales to ever smaller nodes. If variation is not properly or, more importantly, intelligently, addressed, then the margins required to counteract variation grow increasingly large, compromising yield, performance, area, power, and ultimately, cost. We can only mitigate variation by using better fabrication technology, smarter DRM-aware CAD tools, and strict DRC rules, but we cannot eliminate variation or its increasingly large impact. Accordingly, this necessitates that not only do we account for variation at design time, using traditional methods such as corner simulation, Monte Carlo, and CAD tools, but also we must account for variation at run-time, by using smart, efficient, and *in situ* circuits that monitor the various aspects of variation over a chip's operating regime, environmental conditions, and lifetime. The overhead of in-situ variation monitoring circuits is decreasing as technology scaling results in increasing logic density, making it quite a viable method for dynamic run-time variation mitigation. In addition, since variation is present and ever-growing, it cannot be ignored and should be leveraged, especially for applications that require some type of innate randomness or

statistical sampling, such as random number generation and chip ID. These applications then provide a solid basis for further abstraction and leveraging, enabling higher-level applications such as security systems, cryptography, and highly accurate simulation models.

In the introductory chapter, we first discussed the sources of variation and how they impact the circuit designer, and ultimately, the yield of the fabricated chips. We also introduced several previous methods of reducing variation and enhancing yield: multi-Vth design; and body biasing such as reverse, forward, and adaptive body biasing. We then outlined the main contribution of this thesis, and the thesis organization.

Following chapter 1, in chapter 2 we explored in-situ monitoring of the slack paths of a high-performance RISC Alpha core, using a novel, statistically calibrated, all-digital TDC, with an accuracy of 5 picoseconds. The TDC has a small area overhead compared to the rest of the entire core, and its high accuracy allows for extremely adaptive and intelligent ways to adapt the CPU frequency and operation to the performance or power requirements dictated at the moment. The all-digital nature of the TDC also allows it to be calibrated easily, by programming small calibration programs on the actual core and using memory mapping to program and access it. Silicon measurements confirmed that the all-digital TDC meets our target accuracy requirements, and can measure the critical paths accurately and correctly.

In chapter 3, we showed how variation can be exploited for one particularly apt application: on-chip ID generation. The presented system is called OxID. On-chip ID generation is particularly unique in that it has several requirements: randomness (for

security reasons or application reasons); and permanence (one-time programmable). We found that the variation inherent in the oxide breakdown process satisfies both of these requirements of randomness and permanence; this variation was optimally exploited by using dynamic run-time algorithms that automatically tuned our circuitry and control. Our measured silicon results for OxID show that the oxide breakdown process can indeed be used to generate on-chip random IDs that are random in nature, permanent, and robust to PVT. OxID is a new technique and the first chip ID system, known to us, that can generate *permanent* IDs on-chip, at application point, that are immune to PVT effects.

We then discussed in chapter 4 the second application of exploiting the randomness inherent in oxide breakdown: true random number generation. The circuit and system, called OxiGen, were presented, along with the measured silicon results. We found that by measuring the time to gate oxide breakdown and truncating, but not observing, the higher order bits, we can effectively change the random normal distribution to a flat uniform distribution. We can generate a truly and highly random bitstream that passes all fifteen NIST 800-22 suite tests. This was the first time known to us that a hardware true random number generator has passed all fifteen tests with full rigor (appropriately large bitstream sizes for input and many hundreds of streams for full statistical significance, as recommended by the NIST manual), and with no post-processing (direct observation and subsequent manipulation) of the bits in the bitstream required.

Next, in chapter 5, we discussed a novel self-timed transition detection circuit that can be used from super to subthreshold operation. We began with a general discussion of proposed techniques for mitigation of variability in subthreshold design; we then applied

these proposed techniques in a case application to the design of the fully self-timed transition detection circuit. The functionality of this circuit is confirmed in silicon, and simulation data is presented for the circuit.

The work presented in this dissertation can be further extended to more fully explore *in-situ* variation detection and mitigation. Smart, compact algorithms that intelligently adapt the core to the optimal operating conditions can be developed, based on the silicon hardware Razor core and TDC of chapter 2. The TDC of chapter 2 can be further compacted; investigation can be carried out as to minimizing the control overhead of the many digital control and calibration signals required for our current implementation. In particular, a small, compact digital controller can take feedback from the Vernier chain, with a muxing structure for the control signals, to truncate the area, complexity, and power overhead much more. Asynchronous fast-capture TDC with back to back cycle-to-cycle operation (no cycle dead time) has also been theorized and can be further investigated. In addition, while the TDC remains primarily a mixed-signal and analog circuit, in highly digital applications such as that of chapter 2, TDCs that tradeoff resolution and accuracy requirements with area and power (transistor count) must also be considered. Since the TDC is embedded inside a core (or located as close as possible to the critical path signals of interest), the TDC itself changes the design flow and timing of the core. As such, an ultra compact area-minimizing all-digital TDC that trades off resolution and complexity for 1-2 order area improvement has been theorized and should be further investigated.

The OxID and OxiGen works of chapter 3 and 4, respectively, can also be further extended. The basic 3-T cell can be replaced by a 3-chain or 5-chain cell of diode-

connected transistors using all thin-oxide gates; no thick-oxide gates required, as discussed in the chapter 3 conclusion, section 3.7. The chain cells will take their high gate voltage from a globally shared higher-voltage generator. This has been laid-out and the cell/array area shown to be shrunk by 50%. More efficient algorithms for deciding OxID's ID generation stop-point can also be explored; we have explored the two extremes, of full ID monitoring (global algorithm) and small subset monitoring (canary algorithm). Different ways of monitoring, either different physical subsets, or adaptive monitoring that samples different subsets in a specific order, can be investigated. More intelligent stress algorithms for maximizing the random bitstream size of OxiGen's output can also be explored. In addition, OxiGen's throughput is lower compared to other works; while we have offered a simple parallel solution of increasing comparators (and counters), more investigation can be carried out such as sharing a few counters or counter-bits for multiple parallel stressed cells, as the higher-order bits will have the same mean for a specific die.

The chapter 5 work on subthreshold design and variability mitigation techniques can also be further explored. As $V_{DD}$ for systems continue to be scaled down to their energy-optimal point, these kinds of techniques will be required to keep variability in check and to keep systems within operating specifications, namely frequency. Investigation into tradeoffs in subthreshold standard cell sizing (layout and physical design issues such as diffusion area) versus required cell performance should be investigated; currently, standard cells are designed for minimum area and maximum performance in super-threshold. Tradeoffs can be made when considering subthreshold or nearthreshold operation. Furthermore, the specific wide-nor nmos structure and its

inverse wide-nand pmos structure (flattened boolean equation in product-of-sums form and sum-of-products form) of the transition detection circuit and the transistor-to-transistor asynchronous signaling techniques can be further investigated for near and subthreshold applications. The decrease in stages needed in such logic and more localized physical transistors in their layouts may offer the decrease in variability needed to meet frequency and operating specifications in highly scaled near and subthreshold systems.

# REFERENCES

[1] Shekhar Borkar, "Designing Reliable Systems from Unreliable Components: The Challenges of Transistor Variability and Degradation," IEEE Micro, Volume 25, Issue 6, pp. 10-16, 2005.

[2] D. Sylvester, K. Agarwal, S. Shah. "Variability in nanometer CMOS: Impact, analysis, and minimization," Integration: the VLSI journal, pp. 319-339, 2008.

[3] S. Borkar, T. Karnik, S. Narendra, J. Tschanz, A. Keshavarzi, V. De. "Parameter Variations and Impact on Circuits and Microarchitecture," IEEE Design Automation Conference (DAC), pp. 338-342, 2003.

[4] Karnik, T., et al., "Total power optimization by simultaneous dual-Vt allocation and device sizing in high performance microprocessors," IEEE Design Automation Conference (DAC), pp. 486-491, 2002.

[5] Yoonmyung Lee, Mingoo Seok, Scott Hanson, David Blaauw, Dennis Sylvester, "Standby Power Reduction Techniques for Ultra-Low Power Processors," IEEE European Solid-State Circuits Conference (ESSCIRC), September 2008.

[6] Cheng Zhuo, David Blaauw, Dennis Sylvester, "Process Variation and Temperature Aware Reliability Management," ACM/IEEE Design Automation and Test in Europe Conference (DATE), March 2010

[7] D. Fick, N. Liu, Z. Foo, M. Fojtik, J. Seo; D. Sylvester, D. Blaauw. "In situ delay-slack monitor for high-performance processors using an all-digital self-calibrating 5ps resolution time-to-digital converter ," IEEE International Solid-State Circuits Conference (ISSCC), pp. 188-189, 2010.

[8] N. Liu, S. Hanson, D. Sylvester, D. Blaauw. "OxID: On-Chip One-Time Random ID Generation using Oxide Breakdown," IEEE Symposium on VLSI Circuits (VLSIC), June 2010.

[9] N. Liu, N. Pinckney, S. Hanson, D. Sylvester, D. Blaauw. "A True Random Number Generator using Time-Dependent Dielectric Breakdown," IEEE Symposium on VLSI Circuits (VLSIC), June 2010.

[10] S. Hanson, M. Seok, Y-S. Lin, Z. Foo, D. Kim, Y. Lee, N. Liu, D. Sylvester, D. Blaauw, "A Low-Voltage Processor for Sensing Applications with Picowatt Standby Mode," IEEE Journal of Solid State Circuits (JSSC), Invited Paper to the Special Issue on VLSI Circuits, Vol. 44, No. 4, pp. 1145-1155, April 2009.

[11] A. Drake, R. Senger, H. Deogun, G. Carpenter, S. Ghiasi, T. Nguyen, N. James, M. Floyd, V. Pokala, "A Distributed Critical-Path Timing Monitor for a 65nm High-Performance Microprocessor," IEEE International Solid-State Circuits Conference (ISSCC), pp. 398-399, 2007.

[12] K. Woo, S. Meninger, T. Xanthopoulos, E. Crain, D. Ha, D. Ham, “Dual-DLL Based CMOS All-Digital Temperature Sensor for Microprocessor Thermal Monitoring,” IEEE International Solid-State Circuits Conference (ISSCC), pp. 68-71, 2009.

[13] D. Ernst, S. Das, S. Lee, D. Blaauw, T. Austin, T. Mudge, N. Kim, K. Flautner, “Razor: Circuit-Level Correction of Timing Errors for Low-Power Operation”, IEEE, Vol. 24, No. 6, pp. 10-20, November-December 2004.

[14] D. Blaauw, S. Kalaiselvan, K. Lai, W-H. Ma, S. Pant, C. Tokunaga, S. Das, D.Bull, “Razor II: In Situ Error Detection and Correction for PVT and SER Tolerance,” IEEE International Solid-State Circuits Conference (ISSCC), pp. 400-401, 2008.

[15] R. H. Fowler and L. Nordheim, “Electron emission in intense electric-fields,” in Proc. Royal Society of London, vol. 11, pp. 173–181, 1928.

[16] M. Lenzlinger and E. H. Snow, “Fowler-Nordheim tunneling into thermally grown SiO2 ,” J. Applied Physics, vol. 40, pp. 278–283, 1969.

[17] K. Lofstrom, W.R. Daasch, and D. Taylor, “IC Identification Circuit using Device Mismatch,” IEEE International Solid-State Circuits Conference (ISSCC), pp. 372-373, Feb., 2000.

[18] Y.Su, J.Holleman, B. Otis, “A 1.6pJ/bit 96% Stable Chip-ID Generating Circuit using Process Variations,” IEEE International Solid-State Circuits Conference (ISSCC), pp. 406-407, Feb., 2007.

[19] J. Stathis, “Percolation models for gate oxide breakdown,” J. of Applied Physics, Vol. 86, Issue 10, Nov. 1999.

[20] J. Kim, and K. Lee, "Three-Transistor One-Time Programmable (OTP) ROM Cell Array Using Standard CMOS Gate Oxide Antifuse," IEEE Electron Device Letters, Vol. 24, No. 9, Sept., 2003.

[21] P. Candelier, N. Villani, J-P. SchoellKopf, and P. Mortini, "One Time Programmable Drift Antifuse Cell Reliability," IEEE Annual International Reliability Physics Symposium, pp. 169-173, 2000.

[22] H. Ito, and T. Namekawa, "Pure CMOS One-Time Programmable Memory using Gate-Ox Anti-fuse," IEEE Custom Integrated Circuits Conference (CICC), pp. 469-472, 2004.

[23] H-K Cha, I. Yun, J. Kim, B. So, K Chun, and I. Nam, "A 32-KB Standard CMOS Antifuse One-Time Programmable ROM Embedded in a 16-bit Microcontroller," IEEE Journal of Solid-State Circuits (JSSC), Vol. 41, No. 9, Sept. 2006.

[24] M. Isida, and H. Ikeda, “Random Number Generator,” Annals of the Institute of Statistical Mathematics, Volume 8, Number 1, pp 119-126, Nov 1956.

[25] S. Kirkpatrick, and Stoll, “A Very Fast Shift-Register Sequence Random Number Generator,” Journal of Computational Physics, Vol. 40, Issue 2, pp 517-526, 1981.

[26] S.K. Park, and K.W. Miller, "Random Number Generators: Good Ones Are Hard To Find," Communications of the ACM, Vol. 31, Issue 10, Oct 1988.

[27] G. Marsaglia, and A. Zaman, “A New Class of Random Number Generators,” Annals of Applied Probability, Vol. 1, Number 3, pp 462-480, 1991.

[28] M. Matsumoto, and T. Nishimura, “Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator,” ACM Transactions on Modeling and Computer Simulation, Vol. 8, Issue 1, Jan. 1998.

[29] W.H. Payne, J.R. Rabung, T.P. Bogyo, "Coding the Lehmer pseudo-random number generator," Communications of the ACM, Vol. 12, Issue 2, pp 85–86, Feb. 1969.

[30] J. Kelsey, B. Schneier, D. Wagner, and C. Hall, “Cryptanalytic Attacks on Pseudorandom Number Generators,” Fast Software Encryption, Fifth International Workshop Proceedings, Springer-Verlag, pp. 168–188, March 1998.

[31] P. Hortensius, R. McLeod, and H. Card, “Parallel random number generation for VLSI systems using cellular automata,” IEEE Transactions on Computers, Vol. 38, Issue 10, pp. 1466-1473, Oct. 1989.

[32] B. Shackleford, M. Tanaka, R. Carter, and G. Snider, “FPGA implementation of neighborhood-of-four cellular automata random number generators,” Proceedings of the 2002 ACM/SIGDA tenth international symposium on Field-programmable gate arrays, 2002.

[33] M. Tomassini, M. Sipper, M. Zolla, M. Perrenoud, “Generating high-quality random numbers in parallel by cellular automata,” Future Generation Computer Systems, Vol. 16, Issues 2-3, pp. 291-305, Dec. 1999.

[34] P. Tsalides, T. York, and A. Thanailakis, “Pseudorandom number generators for VLSI systems based on linear cellular automata,” IEEE Proceedings of Computers and Digital Techniques, Vol. 138, Issue 4, pp. 241-249, July 1991.

[35] M. Sipper, M. Tomassini, “Generating parallel random number generators by cellular programming,” International Journal of Modern Physics C, Vol. 7, No. 2, pp. 181-190, 1996.

[36] M. Tomassini, M. Sipper, M. Perrenoud, “On the generation of high-quality random numbers by two-dimensional cellular automata,” IEEE Transactions on Computers, Vol. 49, Issue 10, pp. 1146-1151, Oct. 2000.

[37] P. Coddington, “Random number generators for parallel computers,” The NHSE Review, 1996.

[38] B. Jun, and P. Kocher, “The Intel Random Number Generator,” Cryptography Research, Inc., White paper prepared for Intel Corporation, Apr. 1999.

[39] G. Marsaglia, “Random Numbers Fall Mainly in the Planes," Proc. National Academy of Sciences, Volume 61, Issue 1, pp. 25–28. 1968.

[40] Cryptography Research, Inc., "Evaluation of Via C3 Nehemiah Random Number Generator," White paper prepared for Via Corporation, Feb. 2003.

[41] R. Brederlow et al., "A low-power true random number generator using random telegraph noise of single oxide-traps," IEEE International Solid-State Circuits Conference (ISSCC), pp. 1666-1675, Feb 2006.

[42] J. Holleman, S. Bridges, B. Otis, and C. Diorio, "A 3 uW CMOS True Random Number Generator With Adaptive Floating-Gate Offset Cancellation," IEEE Journal of Solid-State Circuits (JSSC), Vol. 43, May 2008.

[43] C. Tokunaga, D. Blaauw, T. Mudge, "True Random Number Generator With a Metastability-Based Quality Control," IEEE Journal of Solid-State Circuits (JSSC), Vol. 43, January 2008.

[44] D. Kinniment, E.G., Chester, "Design of an On-chip Random Number Generator using Metastability," IEEE European Solid-State Circuits Conference (ESSCIRC), pp 595-598, September 2002.

[45] S. Srinivasan et al., "2.4GHz 7mW all-digital PVT-variation tolerant True Random Number Generator in 45nm CMOS," IEEE Symposium on VLSI Circuits (VLSIC), pp 203-204, 2010.

[46] M. Bucci et al., "A high-speed oscillator-based truly random number source for cryptographic applications on a smart card IC," IEEE Trans. on Computers, pp 403-409, April 2003.

[47] C.Petrie, and A.J. Connelly, "A Noise-based IC Random Number Generator for Applications in Cryptography," IEEE Trans. on Circuits and Systems, pp 615-621, May 2000.

[48] F. Pareschi, G. Setti, and R. Rovatti, "A Fast Chaos-based True Random Number Generator for Cryptographic Applications," IEEE European Solid-State Circuits Conference (ESSCIRC), pp 130-133, September 2006.

[49] S. Yasuda et al., "Physical random number generator based on MOS structure after soft breakdown," IEEE Journal of Solid-State Circuits (JSSC), Vol. 39, Issue 8, August 2004.

[50] National Institute of Standards and Technology, Pub 800-22, 2001.

[51] J. Stathis, "Percolation models for gate oxide breakdown," J. of Applied Physics, Vol. 86, Issue 10, Nov. 1999.

[52] J. Kim, and K. Lee, "Three-transistor one-time programmable (OTP) ROM cell array using standard CMOS gate oxide antifuse," Electron Device Letters, pp. 589-591, Sept. 2003.

[53] S. Hanson et al., "Ultralow-Voltage, minimum-energy CMOS," in IBM Journal of Research and Development, Vol. 50, No. 4/5, July/September 2006.

[54] B. Zhai, S. Hanson, D. Blaauw, D. Sylvester, "Analysis and mitigation of variability in subthreshold design," International Symposium on Low Power Electronics and Design (ISLPED), pp 20-25, August 2005.