

# Synthesis Based Design Techniques for Robust, Energy Efficient Subthreshold Circuits

---

A Dissertation

Presented to  
the faculty of the School of Engineering and Applied Science  
University of Virginia

---

in partial fulfillment  
of the requirements for the degree

Doctor of Philosophy

by

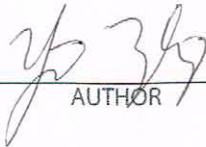
Yanqing Zhang

December

2013

APPROVAL SHEET

The dissertation  
is submitted in partial fulfillment of the requirements  
for the degree of  
Doctor of Philosophy

  
AUTHOR

The dissertation has been read and approved by the examining committee:

Benton Calhoun

Advisor

John Lach

Kevin Skadron

Mircea Stan

Travis Blalock

Accepted for the School of Engineering and Applied Science:



Dean, School of Engineering and Applied Science

December

2013

# **Synthesis Based Design Techniques for Robust, Energy Efficient Subthreshold Circuits**

A Dissertation

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

In partial fulfillment

of the requirements for the degree of

Doctor of Philosophy in Electrical Engineering

by

Yanqing Zhang

December 2013



## **Synthesis Based Design Techniques for Robust, Energy Efficient Subthreshold Circuits**

### **Abstract**

Energy efficiency is increasingly becoming the main concern for many emerging system-on-chip (SoC) applications such as wireless sensor networks (WSNs), body area sensor nodes (BSNs), or portable electronics, which require ultra low power and high energy efficiency. In state-of-the-art situations where the SoC is powered from ambient harvested energy instead of a battery, the constraint of high energy efficiency is put to the extreme in order for the SoC to sustain its operation. Though supply voltage scaling down to near- $(NV_T)$  and sub-threshold( $subV_T$ ), which we refer to together as ultra low voltages (ULVs), has provided drastic quadratic savings in dynamic energy, design of circuits at ULVs still poses important challenges. One of those critical issues is robust design. From a system perspective, for batteryless BSNs that rely on energy harvesting, robust design means maximizing energy efficiency so that the power consumed is less than that harvested, thus ensuring robust, sustained operation of the SoC. Robust design of digital circuits means coping with the acute effects of process variations at ULVs, which are a cause of huge concern. Brute force or conventional methods used in superthreshold to ensure robustness may compromise the goal of energy efficiency at ultra low voltages, or may no longer be sufficient to ensure robustness in this design region. This thesis looks at design techniques to ensure robust design at ultra low voltages, as well as techniques that lower the energy overhead while ensuring a robust design.

For the digital circuits involved in SoCs, ULV operation entails exponentially slower speeds, which not only mean a limit on the throughput available, but also an increase in the significance of leakage current, which may undermine our purpose of energy efficiency. Thus, for SoC architecture, judicious considerations as to the size, amount, type, and communication of modules with respect to energy efficiency must be studied to ensure a deployable design. In this work, we investigate the energy efficiency vs. module platform flexibility design space to answer the question of which type of platform (general purpose processor, FPGA, or ASIC) is most energy efficient in being the main driving force behind digital processing. The exploration of design space also leads to the resulting architecture of a taped-out batteryless BSN SoC.

Increased sensitivity to process variation makes robust timing closure a key challenge at ULVs, and thus it is exceptionally hard for industry to accept ULV designs as future solutions. From a synthesis flow standpoint, this challenge translates to extreme difficulty in timing closure. Straightforward methods to decrease variation and meet timing such as device upsizing are not well suited at ULVs because they compromise the end goal of ultra low power. Conventional methodologies during timing closure that estimate the amount of variation apparent, such as setting a flat timing derate (a value by which all cell delays are multiplied to simulate the effects of variation) for all critical paths, are no longer suitable in the ULV regime. Due to the sensitivity and wide range of  $\sigma/\mu$  of delay across different logic paths in a design, a flat derate will not be able to capture all the correct critical paths in a ULV design. Thus, we propose a modified synthesis flow to ensure the robust design of the digital components of the aforementioned ULV BSN SoC. This thesis explores a method for energy efficient and variation tolerant timing closure using a two-phase, latch based design. We also research and derive a fast method to attain the variation ( $\sigma/\mu$ ) of delay for any logic path in a synthesized design.

The robustness of standard cells comes into question in the ULV regime. Reduced  $I_{on}/I_{off}$  and PVT (process, voltage, and temperature) variations cause stacked transistors to have a much greater chance of failing static noise margin (SNM) and output swing requirements for robust operation. This issue has led designers to explore remedies such as upsizing the static CMOS standard cells, or coming up with a different logic family. However, so far methods to increase robustness come at a power and energy cost, and it is not clear whether other logic families are more energy efficient while ensuring robustness. Thus, we delve into standard cell circuit design for ULVs, and make a systematic comparison of different logic family's energy efficiency under a certain robustness constraint.

# Approval Sheet

This dissertation is submitted in partial fulfillment of the requirements for the degree of  
Doctor of Philosophy (Electrical Engineering)

Yanqing Zhang

Yanqing Zhang

This dissertation has been read and approved by the Examining Committee:

Benton Calhoun

Benton Calhoun, Advisor

John Lach

John Lach, Committee Chair

Mircea Stan

Mircea Stan

Kevin Skadron

Kevin Skadron

Travis Blalock

Travis Blalock

Accepted for the School of Engineering and Applied Science:

James H. Aylor

James H. Aylor, Dean, School of Engineering  
and Applied Science

December 2013



# Acknowledgements

It has been a privilege to work with my Ph.D. advisor, Prof. Benton H. Calhoun, whose work and guidance has inspired and helped me navigate through the course of my graduate degree. I am grateful for his insight and council, and his own work ethic and attitude sets an example for us all. It is my pleasure to thank the members of my thesis committee, Prof. John Lach, Prof. Mircea Stan, Prof. Travis Blalock, Prof. Kevin Skadron, and Prof. Benton Calhoun for their time, many helpful discussions, and suggestions.

I had the most fortunate experience to work closely with Yousef Shakhshere, Alicia Klinefelter, Aatmesh Shrivastava, and Kyle Craig and benefited greatly from their knowledge, insights and interactions. Thank you for your many contributions to this work, and teaching me how to drive a car and play beer pong. I also want to express my appreciation for my colleagues in the RLPVLSI group: Jim Boley, Seyi Ayorinde, Peter Beshay, Patricia Gonzalez, Arijit Banerjee, Divya Akella, He Qi, Yu Huang, Abhishek Roy, Christopher Lukas, Farah Yahya, and Harsh Patel.

Additionally I want to thank: Jiajing Wang, Runjie Zhang, Saad Arrabi, Helen Zhang, Stuart Wooters, Joseph Ryan, Liang Wang, Jerry Qi, Jiawei Huang, Hang Zhang, Xinfei Guo, and Ben Boudaoud. Thank you for the valuable discussions and assistance.

I would like to specifically acknowledge my colleagues from industry who have contributed to this work, especially Tom Gray, John Poulton, and Bill Dally, for the opportunities they provided and enlightening discussions.

Last, but certainly not the least, I wish to thank my beautiful family for their support in getting my degree, namely my parents and my wife.

# Contents

Contents .....	vi
List of Tables .....	ix
List of Figures .....	x
Chapter 1: Introduction .....	1
1.1 Background and Motivation.....	1
1.1.1 Motivation for Subthreshold Circuits .....	1
1.1.2 Motivation for Robust Design.....	2
1.2 Challenges for Robust Design.....	4
1.2.1 Energy Efficient Hardware Selection for ULV SoCs .....	4
1.2.2 Synthesis Flow at ULVs .....	5
1.2.3 Timing Methods and Timing Closure in Subthreshold.....	5
1.2.4 Robust, Energy Efficient Standard Cell Design.....	6
1.3 Summary of Major Contributions and Thesis.....	6
1.4 Thesis Organization .....	8
Chapter 2: Hardware Selection and Synthesis Flow Enabling an Ultra Low Power Battery-less BSN SoC .....	10
2.1 Background .....	10
2.1.1 Energy Efficient Hardware Selection.....	11
2.1.2 Synthesis Flow for Subthreshold .....	13
2.2 Related Work .....	14
2.2.1 State-of-the-art Ultra Low Power BSNs .....	14
2.2.2 Hardware Platform Comparison .....	15
2.2.3 Current Research for Subthreshold Synthesis Flows .....	15
2.3 Solutions .....	16
2.3.1 A Batteryless 19 $\mu$ W Energy Harvesting Body Sensor Node SoC .....	16
2.3.2 Hardware Selection for Flexible DSP Subsystem.....	18
2.3.3 Synthesis Flow for Ultra Low Voltage SoC .....	21
2.3.3.1 Modifications During Cell Characterization .....	21
2.3.3.2 Modifications During Synthesis and Timing Closure.....	23
2.4 Results and Analysis .....	25
2.4.1 Measured Results of Digital Processing Components .....	25
2.4.1 Results of Modified Synthesis Flow .....	29

2.5 Conclusions.....	31
Chapter 3: Variation-Aware Path Timing Computation for Subthreshold Circuits.....	33
3.1 Background.....	33
3.2 Related Work.....	36
3.3 Solution.....	38
3.3.1 Voltage and Temperature Variation in Subthreshold.....	38
3.3.2 Model Derivation in Superthreshold.....	39
3.3.2.1 $\sigma/\mu$ of Single Stage Inverter.....	39
3.3.2.2 Expansion of $\sigma/\mu$ to Path Delay.....	41
3.3.2.3 Expansion of $\sigma/\mu$ to Generic Logic Gates.....	43
3.3.2.4 $\sigma/\mu$ Estimation Method for Superthreshold.....	45
3.3.3 Model in Subthreshold.....	46
3.3.3.1 Device Size in Subthreshold.....	46
3.3.3.2 Slew Dependence in Sub-threshold.....	48
3.3.3.3 Logic Paths and Gates in Subthreshold.....	49
3.3.3.4 Summary of Method in Subthreshold.....	50
3.4 Results and Analysis.....	50
3.4.1 Correct Critical Path Identification.....	52
3.5 Conclusions.....	54
Chapter 4: Timing Closure for Subthreshold Circuits Using a Two-Phase, Latch Based Timing Method.....	56
4.1 Background.....	56
4.1.1 Conventional Timing Closure.....	58
4.1.2 Hold Failure In Subthreshold.....	59
4.1.3 Setup Time Overhead in Subthreshold.....	61
4.2 Related Work.....	62
4.3 Solution.....	65
4.3.1 Hold Time Closure.....	65
4.3.2 Setup Time Closure.....	67
4.3.3 Implementation.....	69
4.4 Results and Analysis.....	70
4.5 Conclusions.....	73
Chapter 5: Logic Family Exploration for Subthreshold Circuits.....	75
5.1 Background.....	75
5.1.1 Overhead of Conventional Static CMOS Libraries.....	76
5.1.2 Alternative Logic Families.....	77

5.1.3 Context of Subthreshold Synthesis .....	81
5.2 Related Work .....	82
5.2.1 Assessment of Superthreshold Logic Families .....	82
5.2.2 Assessment of Subthreshold Logic Families .....	85
5.3 Solution .....	88
5.3.1 Standard Cell Selection and Design .....	89
5.3.2 Robustness Tests .....	92
5.4 Results and Analysis .....	94
5.4.1 Cell Characterization and Cell Energy Efficiency .....	94
5.4.2 Synthesis Nominal Energy Efficiency .....	96
5.4.3 Energy and Speed Variation .....	98
5.5 Conclusions .....	100
Chapter 6: Conclusions .....	102
6.1 Summary of Contributions .....	103
6.2 Contributions In Team Efforts .....	104
6.3 Open Topics .....	106
6.3.1 Chapter 2 (BSN Chip) .....	106
6.3.2 Chapter 3 ( $\sigma/\mu$ Estimation Method) .....	106
6.3.3 Chapter 4 (Two-phase Latch Timing) .....	106
6.3.4 Chapter 5 (Subthreshold Standard Cells) .....	107
Appendix A: List of Acronyms .....	108
Bibliography .....	111

# List of Tables

Table 2.1. Comparison of different hardware platforms. GOPS=Giga operations per second.....	19
Table 2.2. Comparison between DPM and MCU energies.....	20
Table 2.3. Summary of modified synthesis flow for subthreshold SoC tapeout.....	24
Table 4.1. Number of buffers needed to meet hold time per path [45].....	60
Table 5.1. Summary of assessment of different logic families. Red backgrounds mean not suitable for subVT. Yellow means not compared in this work but could be viable with a different scope. Green means compared in this work.....	80
Table 5.2. Qualitative criteria for our logic family study. ....	82
Table 5.3. Table of cells designed for logic family comparison. Each cell can be representative of the behavior of other cells.....	89
Table 5.4. Summary of results of robustness tests. Numbers show minimum size of each gate that passes robustness tests.....	93
Table 5.5. Summary of normalized cost metric (individual cell E-D product) of cells. Some cells that cannot be incorporated in synthesis are given 0.00 values. <i>Dual mode</i> gate numbers include delay and energy of added inverter to retain logic monotony. <i>TX-gate</i> gates include overhead of inverters to create complementary inputs.....	95
Table 5.6. Results of energy efficiency comparison of logic families from synthesis results. ....	96
Table 5.7. $\sigma/\mu$ values of <i>CMOS</i> logic family across different global corners.....	99
Table 5.8. Relative $\sigma/\mu$ of different logic families. All values are normalized to the <i>CMOS</i> case.....	99

# List of Figures

Figure 1.1. Distribution of delay of size X1 inverter in 65nm process across different process corners. Distributions display log-normal tendencies with long end tail and wide spread.....	3
Figure 1.2. Summary of relation between design space, dissertation challenges, and contributions.....	8
Figure 2.1. Example of energy harvester output power. Measured results from an on-body thermal-harvesting experiment with 4×4 cm <sup>2</sup> commercial-off-the-shelf (COTS) thermoelectric generator (TEG). Figure displays roughly 60 μW of output power under nominal conditions [9].....	11
Figure 2.2. Comparison of slew and delay degradation vs. cell output load in super- and sub-threshold. As can be seen, the degradation is more pronounced in subV <sub>T</sub> .....	14
Figure 2.3. System block diagram for the proposed chip comprising the energy harvesting/supply regulation, analog front-end, subthreshold signal processing, and transmitter subsystems [25]. ....	17
Figure 2.4. Diagram of subV <sub>T</sub> processing subsystem. Figure displays high energy efficiency accelerators as well as high flexibility MCU, and sharing of instruction memory (IMEM) [25]. ....	20
Figure 2.5. Example cell SNM failure with NOR2X1 and NAND2X1 gate. Results from MC simulation. Failure of SNM seen with curves not being able to cross each other. The shifting of the VTCs manifest as degraded output voltage levels as well. ....	22
Figure 2.6. Measured system AFib demo experiment using RR extractor and AFib accelerator. Normal and atrial fibrillation heart waveforms from MIT-BIH database [30]. Total chip power in this mode is 19μW from a 30mV input [9]. ....	26
Figure 2.7. Measured energy-delay curves for MCU, RR+AFib, and 30-Tap, 1-Channel FIR [9]. ....	26
Figure 2.8. (Left) Current breakdown for RR extraction experiment. The digital processing subsystem contributes significantly to total drawn current. (Right) Relative energy per operation consumption of different components in digital subsystem, were they to each perform an operation during the same clock period. Memory power (mostly leakage power) is a huge contributor to digital subsystem power [25]. ....	28
Figure 2.9. Average utilization of different cell sizes across 6 synthesized modules in the digital processing subsystem. Cell pruning takes away some size X1 cells, making synthesis tools favor size X2 cells to minimize dynamic and leakage energy. This trend does not change significantly among the 6 modules. ....	30
Figure 3.1. Comparison of $\sigma/\mu$ values between superV <sub>T</sub> and subV <sub>T</sub> regions. SubVT exhibits much more variation. The range of $\sigma/\mu$ is much larger in subV <sub>T</sub> across selected logic paths [33]. ....	34
Figure 3.2. (Left) 10,000 point MC simulation results for delay of a string of inverters at 3 temperatures. Distribution drastically changes across different temperature. Increase of occurrences for T=-20°C end tail (around 80μs) due to low MC sample number. (Right) Average delay of a string of inverters at different supply voltages. Delay varies drastically with V <sub>DD</sub> . ....	39
Figure 3.3. Model vs. MC simulation of single stage inverter pulldown $\sigma/\mu$ across different process corners sweeping device size and load/drive ratio (input slew). Results show close matching between model and MC simulation results. V <sub>DD</sub> =0.9V [33]. ....	40
Figure 3.4. Diagram of circuit for <i>simulation 1</i> , where a chain of inverters' device size ( <i>s</i> ) and load/drive ratio ( <i>c</i> ) are swept uniformly across all stages in the inverter chain. ....	41
Figure 3.5. Extracted $\rho_1$ values from MC simulation on inverter chain pulldown transitions across different operating conditions. $\rho_1$ displays relatively constant value for a given corner [33]. ....	42

Figure 3.6. (Left) Example simulation setup to extract and verify $p_2$ . MC simulations are done sweeping transistor size, load, and stack depth to extract delay through the pulldown/pullup transistor. (Right) Model vs. MC simulation results for NAND2 pulldown $\sigma/\mu$ . Results show close matching [33].	44
Figure 3.7. Summary of $\sigma/\mu$ estimation method. Our method is linear in implementation and has high level of abstraction [33].	45
Figure 3.8. Model vs. MC simulation of selected paths' $\sigma/\mu$ in synthesized processor. Results show close matching [33].	46
Figure 3.9. Model vs. MC simulation of single stage inverter pulldown $\sigma/\mu$ . Device size and input slew (load/drive ratio) are swept. Supply voltage set at $\sim 100\text{mV}$ below threshold [33].	48
Figure 3.10. Model vs. MC simulation error of single stage logic gates' $\sigma/\mu$ in sub-threshold region. Results show good accuracy. Results are for a timing arc with stacked transistors (e. g. pullup path for NOR3 gate). Pulldown stack results presented for AOI21 and OA211 gate. Supply voltage set at $\sim 100\text{mV}$ below threshold [33].	49
Figure 3.11. Model vs. MC simulation of selected paths' $\sigma/\mu$ in subthreshold [33].	50
Figure 3.12. Model vs. MC simulation error of single stage logic gates' $\sigma/\mu$ . Results show good accuracy. Same arc used as in Figure 3.9 [33].	51
Figure 3.13. Critical path delays before and after using the proposed method to estimate $\sigma/\mu$ values and stochastic path delay. Results are for $3\sigma$ yield delay. Figure shows different path priorities based on timing closure method used [33].	53
Figure 4.1. Distribution of delay of size X1 inverter in 65nm process across different process corners. Distributions display log-normal tendencies with long end tail and wide spread [45].	57
Figure 4.2. Conceptual diagram on how performance degrades due to various sources of variation. In an effort to minimize this degradation, extra area and power is incurred as well.	58
Figure 4.3. MC simulation results of hold friendly register's $t_{cq}$ , $t_{hold}$ , and stochastic skew. Stochastic skew arises from unbalanced clock tree by 1 level (a), or balanced tree but clock paths differ by one level (b) or two (c) [45].	59
Figure 4.4. Percent paths failing hold time without hold buffers and percent hold buffer contribution to total block power, displaying the significant overhead of conventional hold buffer insertion [45].	60
Figure 4.5. 16b MAC area across different steps of timing closure. Global corner and OCV contribute greatly to the logic upsizing that results in $>2X$ increase over baseline design area.	61
Figure 4.6. On left, diagram of register based hold failure. 1: Data launched to Q1. 2: Skew causes Q1 to meet setup time $\rightarrow$ hold failure. 3: Desired operation. On right, diagram of two-phase method. 1: Data launched to Q1. 2: Non-complementary phases negate skew even with variation and desired correct operation is ensured. 3: The 'hold shoulder' [45].	67
Figure 4.7. In register designs, logic must be excessively upsized so the delay of the data Q1 on a critical path arrives before edge 1. With time borrowing, logic upsizing can be relaxed because Q1 is allowed to arrive anytime between edge 1 and edge 2.	68
Figure 4.8. Pipe stages 1-2 meet the clock period constraint. Stages 3-4 are critical paths. Without timing borrowing, logic upsizing must be done to lower the stochastic delays of these stages so that they meet the clock period constraint. With time borrowing, stage 3 borrows from stage 4, and stage 4 borrows from stage 5, instead of logic upsizing. Stages 5-9 have less delay than a full clock period, thus allowing time borrowing to recover.	68
Figure 4.9. Implementation flow diagram for two-phase latch based timing method.	69
Figure 4.10. Block diagram of implemented FIR, displaying setup and hold paths.	70
Figure 4.11. Implementation results. (a) Relative energy consumption (latch/register based design). (b) % Hold buffer energy/total block energy for register based designs. (c) Relative energy for logic components (latch/register) [45].	72

Figure 5.1. Conceptual diagram of various logic families [54][56][60][63][66][68][70][71] considered in this chapter, using AND/NAND gate as an example. <i>Bootstrap</i> logic is presented in Figure 5.2.....	79
Figure 5.2. Schematic of simulated bootstrapped inverter. Simulated comparison results on the right, where the supply voltage $V_{DD}=0.3V$ [57][58].....	85
Figure 5.3. Simulation setups for SNM and OSV tests. VTC curves extracted for SNM test. Output logic levels sampled for OSV test. Each gate is loaded by 4 replicas of itself. ....	90
Figure 5.4. Example procedure for standard cell SNM and OSV robustness tests using NAND2 and NOR2 gates. In practice, all gates' VTCs and OSVs are checked.....	93



# **Chapter 1: Introduction**

## **1.1 Background and Motivation**

### **1.1.1 Motivation for Subthreshold Circuits**

In recent years there has been a growing trend for ultra low power, high energy efficiency circuits for such applications as wireless sensor nodes (WSNs) and body area sensor nodes (BSNs). The idea behind these types of applications is to be able to deploy small form factor electronic nodes that attain, process, and in most cases provide feedback data while in their field of operation. For example, WSNs accomplish diverse applications ranging from earthquake monitoring to determining soldier position on the battle field. BSNs—networked body area sensor nodes that continuously capture human physiological data both inside and outside of traditional healthcare settings [1], are revolutionizing the way people receive health care and long term monitoring. The node used in [2] has been a component used in three clinical studies related to movement disorder assessment by providing long term monitoring and data collection.

The key constraint for these applications is node lifetime, since the applications require long term in-the-field operation. Due to the specific application characteristics, it is highly undesirable to frequently switch the nodes' battery. Fortunately, many WSN and BSN applications have a low requirement on the clock frequency. Thus, energy efficient circuits operating in subthreshold become an attractive solution for such nodes. Subthreshold circuits are able to save drastic amounts of dynamic power compared to conventional superthreshold designs [3]. Despite subthreshold circuits being exponentially slower, their speeds are still able to accommodate the application requirements of many WSN and BSN nodes. As the trend of increasing sensor node demand continues to grow where the world will need perhaps more than 10 billion deployable parts to accomplish the future vision of the internet of things [4], subthreshold circuits will increasingly become a vital part in future chips.

### **1.1.2 Motivation for Robust Design**

A critical issue impeding more wide spread acceptance of ULV circuits is their robustness. While circuit robustness can be a broad term, covering design issues such as packaging hardness, soft bit-flip errors, radiation hardness, long term decay problems like NBTI (negative bias temperature instability), and electromigration, robustness within the context of this thesis deals with:

1. Robustness of sustained operation for battery-less, energy harvesting BSN SoCs. Because these SoCs rely on ambient harvested energy, which is generally scarce and subject to environmental fluctuations, it is imperative to ensure architectural decisions lead to the chip consuming less energy than harvested over an extended period of time.

2. Timing closure (the process of checking delays in logic paths and tweaking the design to function correctly at a target frequency) difficulty induced by the acute sensitivity of FET device

current, and in turn gate delay, to process, voltage, and temperature variations (PVT variations), and the energy/power overheads of fixing the induced timing errors. Timing closure robustness is a major issue in sub $V_T$  because subthreshold current is exponentially dependent upon the threshold voltage (eq.1.1). It has been widely accepted that as device sizes continue to scale, the

$$I_D = I_o \frac{W}{L} \exp\left(\frac{V_{GS}-V_T-\eta V_{DS}}{nV_{th}}\right) * \left(1 - \frac{-V_{DS}}{V_{th}}\right) \quad \text{eq. 1.1}$$

main source of process variation is that of RDF (random dopant fluctuation), causing variation in the threshold voltage [5]. Since threshold voltage variation follows a Gaussian distribution [6], gate delays in sub $V_T$  follow a log-normal distribution. Log-normal distributions have long end tails (for example depicted in Figure 1.1), and wider spread (greater  $\mu/\sigma$  value), leading to extreme difficulty in timing closure in sub $V_T$ . Without successful timing closure, it cannot be ensured that the designed circuit can operate as intended. Thus, it is imperative that robust design techniques are adopted, in addition to ensuring high energy efficiency for the applications at hand.

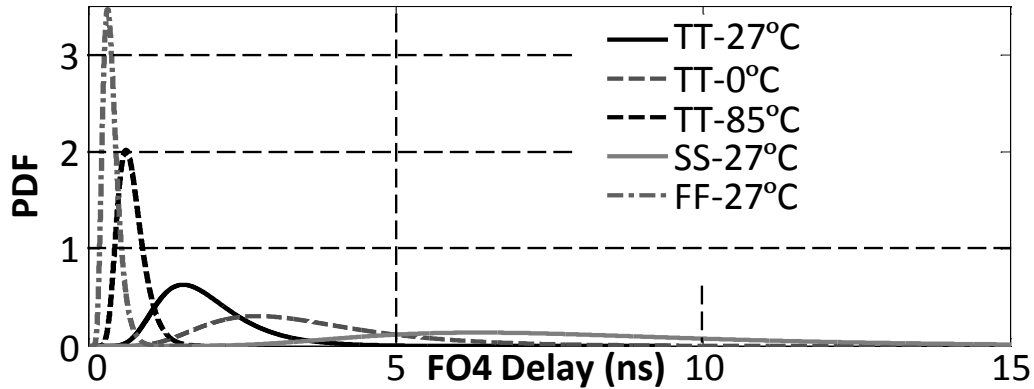


Figure 1.1. Distribution of delay of size X1 inverter in 65nm process across different process corners. Distributions display log-normal tendencies with long end tail and wide spread.

3. Standard cell robustness in terms of static noise margin (SNM) and output swing voltage (OSV). Due to the drastically decreased  $I_{on}/I_{off}$  ratio at ULVs and the acute effects of PVT variations, it is possible that the off current has a comparable or greater value than the on current

in a static CMOS gate. Should this happen, the SNM will deteriorate to the point where the standard cell is no longer capable of performing its logic function. In addition, a deteriorated OSV also has a chance of forcing subsequent cells, which use the reduced output voltage as an input, to the wrong logic level. In short, the issue of standard cell robustness may render a circuit dysfunctional. Thus, it is an important issue.

## 1.2 Challenges for Robust Design

Challenges for robust design include:

1. Energy efficient hardware selection for SoCs.
2. Design of a synthesis flow for ULVs that ensure robust implementation of digital circuits.
3. Design of a timing method that lowers area and energy overhead when ensuring robust timing closure.
4. Derivation of a method to determine correct amount of variation on  $\text{sub}V_T$  logic paths.
5. Design of a standard cell library for  $\text{sub}V_T$  that is energy efficient while ensuring robust operation of the cells.

### 1.2.1 Energy Efficient Hardware Selection for ULV SoCs

SoCs that require prolonged battery lifetime like WSNs and BSNs place great emphasis on optimized energy consumption to meet application characteristics. Given the amount of modules on chip needed to meet often complex processing requirements, the energy cost of digital processing can be high if consideration to the type of processing platform is ignored. Depending on application characteristics, either a flexible microcontroller (MCU) that consumes more energy or a highly energy efficient but application constrained ASIC may be the better choice to

carry the signal processing load of the SoC. Therefore, it is a challenge to design energy efficient architectures and to choose energy efficient modules for the application in question.

### **1.2.2 Synthesis Flow at ULVs**

Given the amount of modules on SoCs, synthesis flows are employed to streamline the design of digital blocks, thus enabling the billion transistor count chips apparent in industry today. However, synthesis constraints change in the ULV regime. For example, cell robustness becomes a greater issue and guardband measures must be more stringent because of increased sensitivity to PVT variations. Thus, it is a challenge to design a synthesis flow suitable for ULV designs.

### **1.2.3 Timing Methods and Timing Closure in Subthreshold**

Timing closure is imperative for any digital circuit to operate correctly. The long end tails of the log-normal distributions make logic paths prone to setup failure. PVT variations also give rise to heightened failure rate for hold violations, as contamination delays can be much faster, clock slew much slower, clock skew much higher, and hold robustness of registers much less. Given a determined set of standard cells, often times the straightforward method for easier timing closure is to upsize the cells (only use higher drive strength cells) so the variation ( $\sigma/\mu$ ) is less [6]. This will incur an area and energy overhead. In addition, the straightforward solution to fixing hold time errors, hold buffer insertion, also incurs area and energy overheads. The added overheads may compromise the circuits' end goal of high energy efficiency in  $\text{sub}V_T$ . Hence, it is a challenge to come up with a timing method that ensures robust timing closure that may lower the overheads of conventional closure methods.

In addition, the increased amount of  $\sigma/\mu$ , as well as delay distribution being non-Gaussian in  $\text{sub}V_T$ , render traditional timing closure flows, which utilize flat timing derates [7] and Gaussian

distribution properties, during synthesis unusable for  $\text{sub}V_T$ . So, methods of determining the correct  $\sigma/\mu$  value for  $\text{sub}V_T$  logic paths must be explored for robust timing closure.

### **1.2.4 Robust, Energy Efficient Standard Cell Design**

Standard cells are the building blocks of large integrated digital circuits, functioning as the Lego blocks during the synthesis process. While the design process of standard cells for superthreshold is well established and understood, there is only emerging research [8] regarding standard cell design in  $\text{sub}V_T$ . At the centerfold of the challenge is that due to increased sensitivity to variation and different balance of P- and N-type devices in  $\text{sub}V_T$ , robustness of standard cells in terms of static noise margin (SNM) and reduced output swing voltage (OSV) [8] becomes a key design metric. Thus, standard cell transistor sizing, as well as logic family exploration, must be researched with robustness as the key metric as opposed to the traditional PPA (power, performance, and area) optimization approach used in superthreshold.

## **1.3 Summary of Major Contributions and Thesis**

This thesis addresses the overarching issue of maintaining ultra low power and high energy efficiency while ensuring robustness across different levels of abstraction (architecture, timing/logic, and transistor).

To address the issue of robust, sustained batteryless operation using energy efficient hardware selection, we perform a multi-platform analysis between ASIC accelerators, FPGAs, and GPPs (general purpose processors). Our simulation results show  $\sim 1000X$  improvement in energy efficiency from ASIC accelerators over GPPs. This conclusion leads us to the implementation of a body area sensor node (BSN) chip architecture with dedicated power manager microcontroller (MCU) and accelerators for energy efficient processing. The BSN chip

was taped-out and measurement data supports our decision on efficient hardware selection. A dedicated synthesis flow for ULV SoC design was also implemented for the tapeout, and its validity was verified through test chip measurement results, which show sustained robust operation across all 10 chips tested.

We propose a fast tool to compute the variation ( $\sigma/\mu$ ) of delay for any logic path in a sub $V_T$  synthesized design to deal with the challenge of determining correct  $\sigma/\mu$  values. We demonstrate the importance of using the fast variation estimation method to identify critical paths in sub $V_T$  designs, as the logic path with longest nominal delay may not have the greatest stochastic delay ( $\mu+x\sigma$ ). The proposed method does not require deep understanding of device physics, prior knowledge of the design, or extensive Monte Carlo simulation, and it provides good accuracy with less than 11% error.

To alleviate the overhead of ensuring robust timing closure, we propose a two-phase, latch based timing method. Through simulation results we find that compared to conventional hold buffering, our solution saves up to 37% (at  $6\sigma$  yield) in energy per operation and allows for post tapeout hold time correction. Replacing registers with latches also permits time borrowing, which we show can save up to 46% in energy per operation when used for setup time closure.

Finally, to address the question of which logic family is best suited for sub $V_T$  synthesis, we perform a logic family analysis covering static CMOS, dynamic, transmission gate based, and differential cascade voltage switch (DCVS) families. Our simulation results make the case that static CMOS should be the logic family of choice under the same SNM and OSV robustness constraints for standard cells. Furthermore, within the variants of static CMOS logic (multi-threshold CMOS, dynamic threshold CMOS, etc.), NAND-only (*NONLY*) CMOS provides the

lowest energy per operation at the typical process corner, and has the lowest energy and delay variation while accomplishing the same frequency as the conventional static CMOS counterpart.

A summary of how the challenges and contributions relate within the digital circuit design space is given in Figure 1.2. It also shows which chapter each contribution lies in.

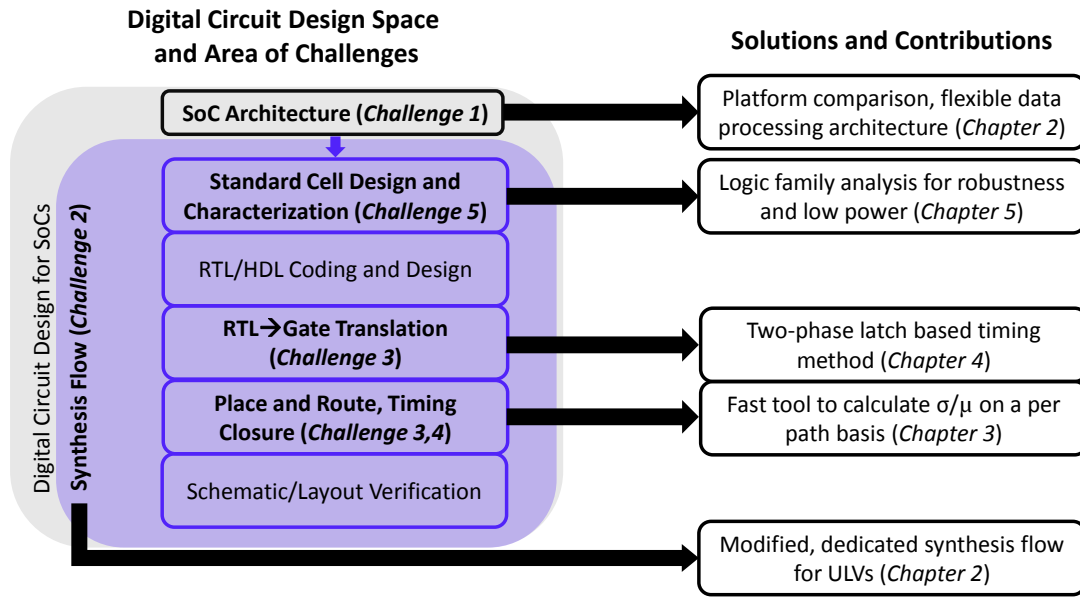


Figure 1.2. Summary of relation between design space, dissertation challenges, and contributions.

## 1.4 Thesis Organization

The remainder of this thesis is organized as follows.

Chapter 2 presents a taped-out BSN chip accomplishing potentially perpetual operation from energy harvesting. A digital processing platform analysis and sub $V_T$  synthesis flow are presented as two important contributions enabling the success of the chip. Based on measured results, the BSN is capable of batteryless operation due to the ultra low power levels achieved. A treatment is also given as to the impact of the platform analysis and synthesis flow on the overall BSN chip.

Chapter 3 presents the fast tool to calculate the  $\sigma/\mu$  of logic paths. The chapter gives an overview of how on chip variation (OCV) is dealt with conventionally using a flat timing derate for critical paths, and shows how this method is unsuitable for sub $V_T$  voltages. Typical values of



$\sigma/\mu$  are also characterized. The advantages of the proposed tool are displayed when we show our tool relies only on a few easily extractable parameters from a few simulations, and is capable of correctly tracking the true critical paths in synthesized designs based on stochastic delays.

Chapter 4 describes the two-phase clock, latch based timing method that lowers the energy overhead of ensuring robust timing closure in the face of OCVs. An analysis of the energy overheads, including logic upsizing and hold buffer insertion, incurred is given. We show through simulation data the advantages of using the two-phase, latch based method, where less logic upsizing and no hold buffers are needed. Finally, we describe how to implement the two-phase, latch based method in standard synthesis flows, and comment on potential overheads of the implementation.

Chapter 5 explores different logic families' behavior in  $\text{sub}V_T$  and their suitability for synthesis in the ULV region. We argue that that, in  $\text{sub}V_T$ , robustness should be a key metric to design around, in addition to the conventional PPA metrics for standard cells. We analyze the advantages and disadvantages of different logic families and especially their robustness in  $\text{sub}V_T$ . We show through simulation data that under the same robustness and frequency constraints, NAND-only static CMOS logic provides the lowest energy per operation, leakage current, and delay and energy variation.

Finally, Chapter 6 concludes this thesis. This chapter points out areas for future research work and improvements for the design techniques described in this thesis.

# **Chapter 2: Hardware Selection and Synthesis Flow Enabling an Ultra Low Power Battery-less BSN SoC**

## **2.1 Background**

There has been a growing interest in body sensor nodes (BSNs), as they promise to provide significant benefits to the healthcare domain by enabling continuous monitoring, actuation, and logging of patient bio-signal data, which can help medical personnel to diagnose, prevent, and respond to various illnesses such as diabetes, asthma, and heart attacks [1]. Though they show great potential, BSNs have many design challenges that impede their widespread adoption including node operating lifetime, small form factor for wearability, and affordable cost. One of

---

This chapter is based off the published paper titled “A Batteryless 19  $\mu$ W MICS/ISM-Band Energy Harvesting Body Sensor Node SoC for ExG Applications” [YQ9].

the most critical issues is node lifetime. In many applications, such as long-term monitoring of chronic illnesses, limited battery lifetimes severely undermine the deployment of BSNs, since the required node operating lifetime is effectively indefinite. Supplying the node with sufficient energy over a long lifetime poses a challenge. A large battery increases the form factor of the node, making the node unwearable or uncomfortable, while a small battery requires frequent changing and reduces wearer compliance. Energy harvesting from ambient energy sources, such as thermal gradients or mechanical vibrations, potentially provides indefinite lifetime. To eliminate battery changing, nodes can operate solely from energy harvesting instead of using a battery, although this introduces new challenges. For example, the full system must consume less energy than the amount harvested, high power components such as the transmitter must be heavily duty-cycled, and the node must cope with time varying harvested energy profiles.

### 2.1.1 Energy Efficient Hardware Selection

Since a drawback of energy harvesters is their limited power output (Figure 2.1), it is difficult to ensure a batteryless BSN's robust operation, in the sense of sustaining that the chip consumes less energy than harvested. As can be seen from Figure 2.1, only  $60\mu\text{W}$  of power can be harvested under typical conditions, and the amount supplied to the chip will be even lower

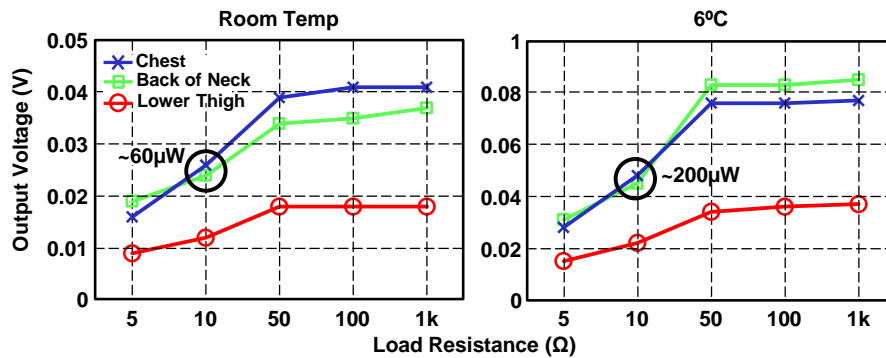


Figure 2.1. Example of energy harvester output power. Measured results from an on-body thermal-harvesting experiment with  $4 \times 4 \text{ cm}^2$  commercial-off-the-shelf (COTS) thermoelectric generator (TEG). Figure displays roughly  $60 \mu\text{W}$  of output power under nominal conditions [9].

(<30 $\mu$ W) due to low efficiency of the chip's boost and DC-DC converter. Therefore, it is important, when thinking about the architecture of the BSN SoC and the variety of components on it, to make judicial decisions to which components to include so that energy efficiency is maximized while still meeting the throughput and processing capability requirements of the application. Where in economics we want to 'make every dollar count', for the SoC we wish to 'make every pJ count'.

Thus arises the question of determining how to maximize energy efficiency using hardware selection given a variety of application based needs. The selection spreads from the highly flexible but lower energy efficiency general purpose processors (GPPs) to the highly energy efficient but non-flexible ASIC accelerator modules. GPPs exhibit poor energy efficiency due to the overhead of fetching and decoding the instructions that are required to perform a given operation in the datapath. For low power embedded applications like BSNs, general purpose computation is generally performed in fairly simple microcontrollers [10][11][12].

The most efficient hardware is hardwired to do its specific task or tasks (e.g., ASIC). ASICs achieve very efficient operation, but they can only perform the function for which they were originally defined. Examples of hardwired implementations in sub-threshold circuits include [13][14]. The ASIC implementation of a JPEG co-processor in [14] consumes 1.3 pJ/frame for VGA JPEG encoding.

Microprocessor operations are largely inefficient, as we described above. Field Programmable Gate Arrays (FPGAs) are reprogrammable hardware that provide an intermediate choice between ASICs and processors in terms of flexibility and efficiency. An FPGA is configured to act like specific hardware, similar to an ASIC, but the configuration can be changed an arbitrary number of times. The cost of this flexibility is that FPGAs consume 10~100

times more energy than an ASIC due to energy overhead from interconnects, which may account for 85% of the total energy consumption. Most commercial FPGAs target high performance applications to compete with processors, but a sub $V_T$  FPGA [26] demonstrates that custom FPGA implementations can offer a good tradeoff for flexibility and energy efficiency for energy constrained applications like BSNs.

Our architectural design decisions will be informed from the study on the different processing platforms mentioned, **which addresses challenge 1**, and in this way, we will ensure minimal energy overhead of digital processing on the BSN, which contributes to the robust, sustained operation of the batteryless BSN.

### 2.1.2 Synthesis Flow for Subthreshold

Given the abundant amount of digital modules needed on an SoC, a synthesis flow that will ensure a robust, functional design is vital to the correct operation of the BSN SoC. A synthesis flow provides constraints to the EDA tools that will translate high level RTL language to final transistor and layout design. Whilst synthesis techniques are well understood and established for super-threshold designs, there is only emerging research to its improvement for sub $V_T$  designs [16]. Within the many steps of a synthesis flow, some are heavily affected by the supply voltage region change to subthreshold. For example, during cell characterization, some cells conventionally utilized in super-threshold may not be usable or desirable in sub $V_T$  due to the increased imbalance of P/N type device current ratio, which in turn degrades cell robustness, delay, and cell loadability. In cases where the amount of OCV has not been fully characterized, increased amount of guardbands must be set during timing closure in anticipation of the increased sensitivity to variation that sub $V_T$  circuits exhibit. Also during timing closure, the corners at which to check timing will change, mainly due to the fact that sub $V_T$  devices run

slower at lower temperatures, which is contrary to super-threshold, where the slowest temperature corner is usually at extreme highs. Lastly, extra emphasis to controlling slew must be observed, as slew degrades more quickly with load in sub $V_T$  due to low drive current when compared to super-threshold design (Figure 2.2). Therefore, we will modify the synthesis flow so it is suitable for design of sub $V_T$  modules, **thus addressing Challenge 2**.

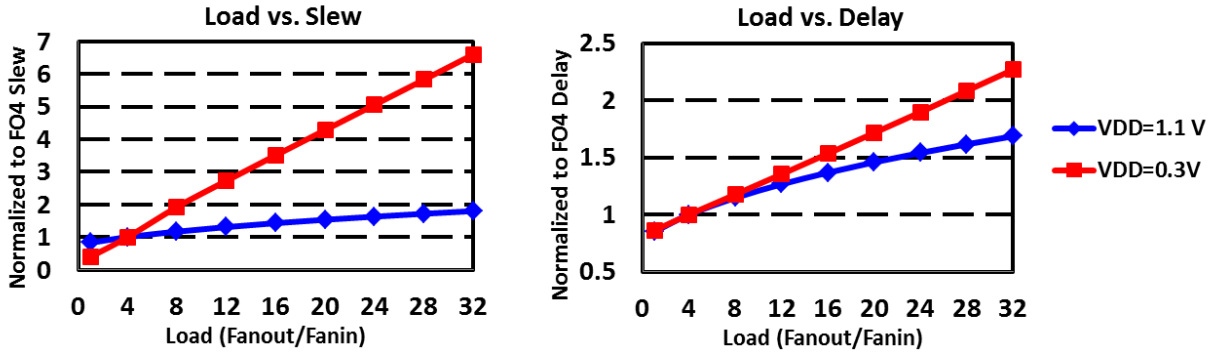


Figure 2.2. Comparison of slew and delay degradation vs. cell output load in super- and sub-threshold. As can be seen, the degradation is more pronounced in sub $V_T$ .

## 2.2 Related Work

### 2.2.1 State-of-the-art Ultra Low Power BSNs

Recent advances in ultra-low power chip design techniques have enabled a surge in the design of a new generation of body-worn devices for health monitoring. Some of the state of the art examples of such designs include [17]-[21]. In [17], an ECG acquisition and processing SoC with a 3-channel analog front-end (AFE) and generic DSP components is presented. The on-chip processing is assigned to a powerful processor with semi-custom execution units. The digital processing power consumes >30% of the chip's total power, limiting the minimum power to 31.1 $\mu$ W, which is too large to support energy harvesting. The system in [19] integrates several chips with solar cells and a battery to accomplish near-perpetual operation for measuring intraocular pressure. The system consumes 3.3fW/bit at 400mV when taking one measurement

every hour. The system accomplishes energy harvesting as the main source of power by operating at an extremely low sampling rate with limited digital processing on chip. Even as such, the choice to use a CORTEX GPP leads to digital power being >25% of the total chip's power. Furthermore, though state of the art, these two systems chose to select GPP-like digital modules to perform the on chip digital processing, and this limits these systems' capabilities to become a complete wireless, flexible, easily deployable BSN node that supports energy harvesting. These observations further our assertion that careful digital platform analysis is needed to reach our goal of a wholesome, wireless, BSN node with power management and energy harvesting.

### **2.2.2 Hardware Platform Comparison**

The topic of hardware platform comparison and the tradeoff between flexibility and efficiency in hardware is well known and very prominent in a comparison of conventional hardware paradigms [22][23]. In [23], the authors achieve an energy efficiency increase factor of 6 by mapping algorithms to the most efficient platform for execution. In this way, processor components exhibit their advantage in control flow, FPGAs exhibit their reconfigurability, and ASICs exhibit their powerful processing capabilities. The methods described in these papers can guide us in our own comparison, as how the comparison would translate to the sub $V_T$  regime remains to be seen.

### **2.2.3 Current Research for Subthreshold Synthesis Flows**

Finally, little research has been done to the synthesis flow optimization in sub $V_T$ . [16] concludes that it is not a good idea to synthesize in super-threshold and scale down to sub $V_T$  supply voltages, lest an energy efficiency decrease of >30% be incurred. This suggests that cell re-characterization should be done for sub $V_T$  designs to enable sub $V_T$  synthesis, a conclusion that coincides with one of our goals.

## 2.3 Solutions

To better understand the context of our platform comparison and synthesis flow, we first present the over-arching results of our successful tapeout of a batteryless BSN SoC for flexible biomedical applications. Afterwards, we present our platform and synthesis solutions, and how they contributed to the completion of the chip.

### 2.3.1 A Batteryless 19 $\mu$ W Energy Harvesting Body Sensor Node SoC

We utilize recent advances in energy harvesting, low voltage boost circuits, dynamic power management, subthreshold processing, bio-signal front-ends, and low power RF transmitters to realize an integrated reconfigurable wireless BSN SoC for ECG (electrocardiogram), EMG (electromyogram), and EEG (electroencephalogram) applications with autonomous power management for completely battery-free operation [25]. A specific application our SoC targets is atrial fibrillation detection (AFib) [24]. Other applications specifically implemented include heart rate extraction (RR extraction), digital bio-signal filtering, and energy band extraction (envelope detection), which can be expandable to other applications for ECG, EMG, and EEG. The SoC can run indefinitely from energy harvested from body heat while worn, and potentially decreases cost by having high integration and targeting a wide range of bio-electric sensing applications.

To achieve flexible data acquisition and processing while operating the node solely from harvested energy, we propose a system architecture, illustrated in Figure 2.3, which comprises four subsystems. First, the energy harvesting/supply regulation section boosts a harvested supply input as low as 30mV up to a regulated 1.35V using an off chip storage capacitor. It provides



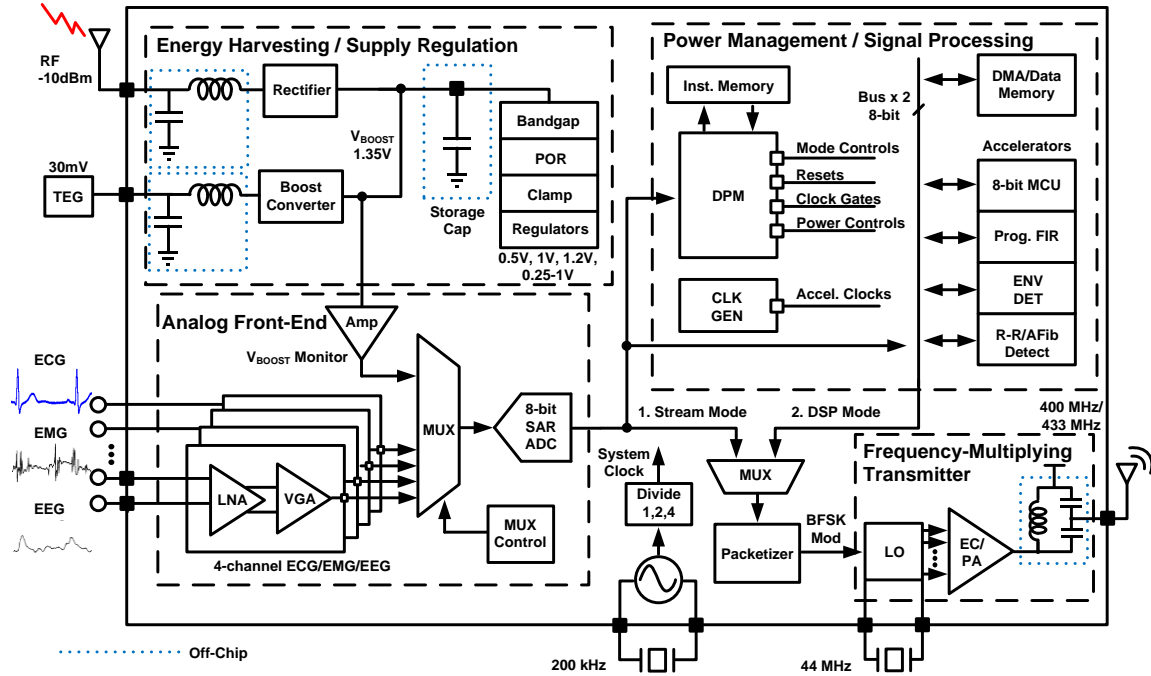


Figure 2.3. System block diagram for the proposed chip comprising the energy harvesting/supply regulation, analog front-end, subthreshold signal processing, and transmitter subsystems [25].

five regulated voltage supplies to the rest of the chip, and generates a bandgap reference. Second, the four-channel analog front end (AFE) subsystem provides bio-signal acquisition with programmable gain and sampling rate, amplifying bio-signals as low as a few  $\mu\text{V}$ 's while consuming  $<4\mu\text{W}/\text{channel}$ . Third, the acquired data is sent to a sub $V_T$  digital processing subsystem that also performs mode control and power management (including power/clock-gating of blocks and dynamic voltage scaling (DVS)) based on the available energy on the storage capacitor. The digital section includes a custom digital power management (DPM) processor, general purpose microprocessor (MCU), programmable FIR, 1.5kB instruction SRAM/ROM, 4kB data memory FIFO, and dedicated accelerators for ECG heart rate (R-R) extraction, atrial fibrillation (AFib) detection, and EEG band energy calculation. It is in this subsystem where the results of our platform comparison and synthesis flow are put to implementation use. The DPM is responsible for power management, node control, data flow management, and overseeing all processing on-node. Finally, a sub-mW 400/433 MHz

MICS/ISM band frequency-multiplying transmitter (TX) performs BFSK transmission up to 200 kbps. The TX is intelligently duty-cycled to achieve low average power consumption.

### **2.3.2 Hardware Selection for Flexible DSP Subsystem**

In the digital signal processing subsystem, we are faced with the challenge of implementing several specific target applications, as well as maintaining flexibility to expand the chip's application space for other algorithms. All of these chip requirements must be met under a stringent energy and power budget to ensure batteryless operation. Thus, to better understand the energy vs. flexibility tradeoff, we propose a study of three platforms: GPP, FPGA, and ASIC accelerator. To put this comparison in fair context with ultra low power SoCs for biomedical purposes, we implemented the same heart rate extraction algorithm on all three. We also manually implemented all three platforms in the same technology and used the circuit optimization techniques available to us for a custom energy efficiency implementation. We used a state-of-the-art ULV design for the FPGA [15]. We hand optimized the assembly code for the GPP, a synthesized 8b RISC processor [26], and hand optimized the verilog circuit model to ensure we had accomplished the most energy efficient implementation for each platform. We then performed Spice simulation of our circuits and verified correct functionality of execution of our RR algorithm, and extracted our key metrics of energy/op, delay, and # of instructions per processed sample.

The results of our experiment are presented in Table 2.1. The key observation is that there is a drastic improvement in efficiency (>100X) between GPPs and FPGA/ASICs. Therefore, it makes sense to assign the processing duties to ASIC platforms wherever possible, while using GPPs strictly for control or rarely occurring subroutine operations. This is the key conclusion that our BSN chip utilizes.

Therefore, it is without a doubt we chose to implement the AFib as a hardware accelerator. With the massive amount of energy efficiency achievable from ASICs in  $\text{sub}V_T$ , we look to expand the role of accelerators on our chip further. We chose to have ASICs perform commonly occurring functions in the context of our SoC application requirements. An FIR, R-R extraction, and Envelope Detector are added as hardware accelerators. These operations can take several instructions over many clock cycles to complete using a GPP, consuming a large amount of energy and latency. Though hardware accelerators process data in very specific ways, their functions fit within the flexible datapath subsystem as many applications will require signal filtering, heart rate extraction (for ECG applications), or energy band calculations (envelope detector for EEG and EMG applications).

Table 2.1. Comparison of different hardware platforms. GOPS=Giga operations per second.

	<b>Energy per Instruction</b>	<b>Energy per Sample</b>	<b>Delay per Sample</b>	<b>Achievable data rate</b>	<b>GOPS / W</b>
GPP	2.62 pJ	210 pJ	8 us (80 cycles)	125 kHz	4.76
FPGA	N/A	2.22 pJ	94.5 ns (1 cycle)	10 MHz	450
ASIC	N/A	0.23pJ	6.18 ns (1 cycle)	150 MHz	4348

Where ASICs excel in energy efficiency, they fail to provide us with a solution for needed flexibility in our flexible processing datapath. To remedy this, we choose to also include an 8b GPP in the subsystem. Most of the time, it is power gated, but it is turned on and is programmable to implement the rarely occurring subroutine. Another advantage GPPs provide us is data and power flow control, which we will need in order to organize our numerous accelerators. However, given the results in Table 2.1 and knowing the low energy efficiency GPPs exhibit, we delegated these responsibilities to a custom designed processor, the digital power manager (DPM). The DPM executes instructions from a 1.5kB instruction memory and provides a lower energy alternative to using generic MCUs for controlling the node (Table 2.2).

Table 2.2. Comparison between DPM and MCU energies.

DPM Operation	DPM Energy (pJ)	MCU Equivalent (pJ)
Energy Efficient NOP	0.7 pJ	1.46 pJ
Control Signals	2.8 pJ	2.92 pJ
Subroutine Commands	2.9 pJ	4.38 pJ

In order to lower the overhead of instructions SRAM, the DPM and GPP MCU share the same bank of memory. A multiplexer steers each instruction read from the instruction memory to either the MCU or DPM based on a special code word. When the MCU is executing instructions, the DPM automatically goes into a low power sleep mode. When the DPM is executing instructions, the MCU is either turned off or clock gated to save state. In this way, we retain the energy efficiency of the DPM as a chip controller and the generic flexibility of the MCU without requiring extra instruction memory space. The MCU's instructions are programmed at the same time as the DPM during the chip's pre-deployment.

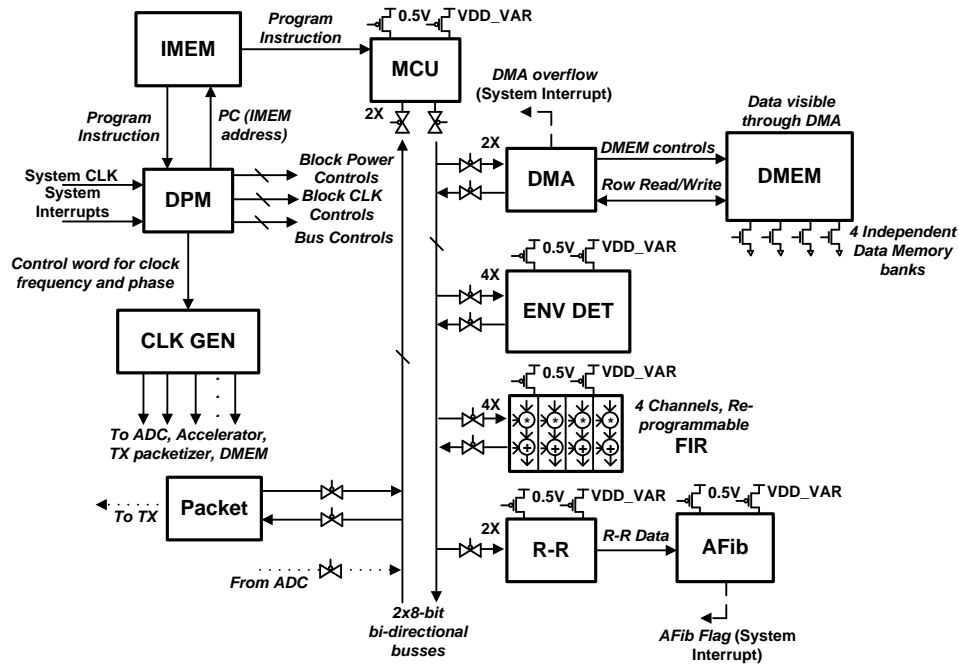


Figure 2.4. Diagram of sub $V_T$  processing subsystem. Figure displays high energy efficiency accelerators as well as high flexibility MCU, and sharing of instruction memory (IMEM) [25].

Figure 2.4 shows the resulting subthreshold DSP subsystem, partially determined from our platform comparison and analysis. The chip can process data flexibly with the MCU, use the highly efficient hardware accelerators, or cascade accelerators with MCU processing.

### **2.3.3 Synthesis Flow for Ultra Low Voltage SoC**

#### **2.3.3.1 Modifications During Cell Characterization**

Having determined the architecture of the subsystem, the next issue to tackle is its implementation. Using the results of [16], we determine that we will synthesize the modules in  $\text{sub}V_T$ . Thus, we need to re-characterize the standard cells at our  $\text{sub}V_T$  operating voltage of 0.5V. Given the results of [12], we need to be aware of cell robustness. Also during cell characterization, we must be aware of the faster degradation of slew and gate delay with increasing output load (Figure 2.2). We will have to pay more attention to characterization at lower temperatures, since the slowest corner in  $\text{sub}V_T$  occurs at low temperature extremes. In contrast, we can relax the high temperature constraint for cell characterization and eventually timing closure, since  $\text{sub}V_T$  circuits are slow and dissipate much less heat than in super-threshold.

We start our modified synthesis flow with cell characterization. The first modification we make is the global corners chosen for characterization. We establish that our typical corner is TT:0.5V:27°C (representing typical-typical MOS corner, typical supply voltage of 0.5V, and typical room temperature 27°C). With consideration to setup time, we must identify the slowest global corner. Whilst the slowest corner can be at extreme high temperatures due to carrier mobility decreasing (because of lattice scattering effects), the slowest corner in  $\text{sub}V_T$  is at extreme low temperatures where the effects of an increased threshold voltage at low temperatures dominate the speed. In addition, though  $\text{sub}V_T$  circuits do now draw large amount of currents to cause significant IR drop directly from the supply, our architecture (Figure 2.3)

compels that PDVS headers will be used to incorporate power-gating and DVS capabilities. With PDVS header design, it can be expected to have a worst case -10% degradation in the virtual rail supply voltage [27]. Thus, a characterization at -10% supply voltage degradation should be done. The slowest corner is SS:0.45V:0°C. Hold time checking needs to be done at various corners, as it is not intuitive how the different hold parameters of  $t_{cq}$  and  $t_{hold}$  will change relative to each other. It is, however, certain that stochastic skew will be greater at lower temperatures from Figure 1.1. Thus, more corners should be characterized at low temperatures for hold time. We chose to characterize at FF:0.5V:45°C, SF:0.45V:0°C, and FS:0.45V:0°C for hold time.

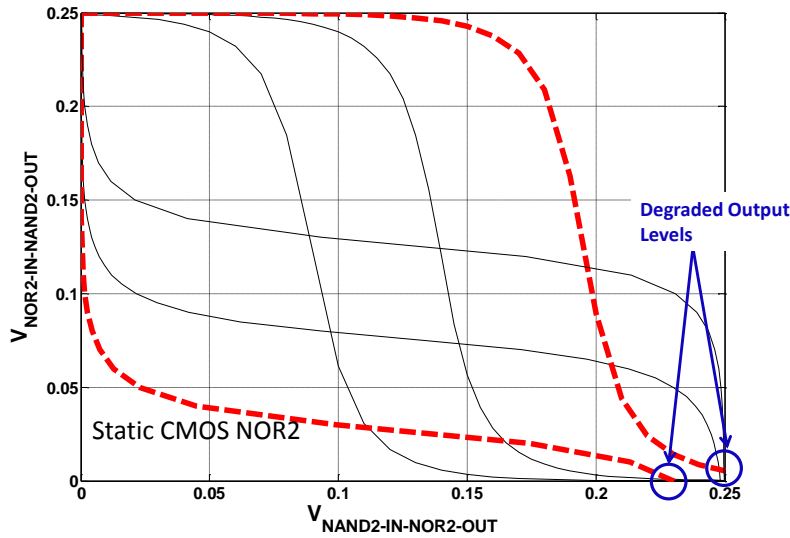


Figure 2.5. Example cell SNM failure with NOR2X1 and NAND2X1 gate. Results from MC simulation. Failure of SNM seen with curves not being able to cross each other. The shifting of the VTCs manifest as degraded output voltage levels as well.

The next issue of output load control to limit the decrease in slew and gate delay compels us to modify the constraints placed during the implementation of cell characterization. “Constraints”, from a cell characterization perspective, means the user defined input slew and output load for each timing arc that will comprise the timing lookup-up table [28]. Namely, the input slews must have a wider coverage when compared to the input slew constraints of super-

threshold due to its quick degradation. Thus, the input slews cover a range of 1-16 FO4 slews to ensure accurate characterization of the cells.

The final modification made during cell characterization is cell pruning. Due to degraded  $I_{on}/I_{off}$  ratio in  $subV_T$ , cell robustness becomes an issue. Given the immense number of gates in a full standard cell library (commonly >600), consideration must be given as to how to decrease the simulation time to prune out cells that have low robustness yield. Therefore, we chose to use the method of performing OSV analysis with Monte Carlo (MC) simulation before cell characterization. SNM analysis is not done, since we identify that it is essentially the output voltage level that will determine the gate's robustness, as the shifting of VTC curves in SNM analysis will ultimately manifest itself as degraded OSV (Figure 2.5). In addition, we chose to run MC simulation with the option of having the global corner also determined by MC variations, as opposed to having a set global corner, and only local variations. In this way, we limit the number of MC simulations to one runset, as opposed to having 5 runsets, one for each determined global corner (TT:0.5V:27°C, SS:0.45V:0°C, FF:0.5V:45°C, SF:0.45V:0°C, and FS:0.45V:0°C). Cells are pruned out of the standard cell library and do not enter cell characterization if they fail OSV analysis by a hard constraint of 5% of nominal  $V_{DD}$  (25mV for logic '0', 475mV for logic '1') to ensure cell robustness. Though this is a coarse and conservative approach to ensure cell robustness, it is quick and easy to implement. Instead, we defer to Chapter 5 of this thesis on a more comprehensive approach.

### **2.3.3.2 Modifications During Synthesis and Timing Closure**

A unique constraint that must be added for  $subV_T$  synthesis and timing closure is to relax the maximum delay (make this constraint a larger number) allowed for each cell. This is again due to the quick degradation of slew with increasing load in  $subV_T$ . Without relaxing this constraint,

synthesis and timing closure tools tend to increase the module's area drastically. What happens is during the speed optimization step, the synthesis tools realize the great slew degradation. In order to optimize speed and keep slew in check, the tools will choose to create many duplicate logic paths, which will decrease the load, and in turn slew, of each logic gate. However, this decision by the synthesis tool drastically increases the area and energy of the module with limited improvement in speed. Relaxing the gate maximum delay constraint gets rid of this phenomena, and the synthesis tools interpret the cells as loadable again.

The other modified constraint for synthesis and timing closure is the conservative guardbands added to robustly close timing in the face of OCVs. We applied guardbands by running MC simulations on a string of size X1 inverters whose logic depth equals that of the synthesized critical path to determine a conservative  $\sigma/\mu$  value. The value extracted from this simulation is conservative because  $\sigma/\mu$  decreases with larger sized gates [6]. After the  $\sigma/\mu$  value is attained, a guardband value of  $\exp(3*(\ln\sigma^2/\mu^2+1)^{0.5})$ , which represents the  $3\sigma$  yield added stochastic delay if the path were constructed of all size X1 inverters according to log-normal distribution equations, is added to the closure of all logic paths. Once again, though this method is pessimistic and conservative, it is quick and easy to implement and ensure robust timing closure. We defer a comprehensive treatment of determining  $\sigma/\mu$  values and robust timing closure to Chapter 3 of this thesis.

Table 2.3 shows a summary of the modifications we made for the synthesis flow.

Table 2.3. Summary of modified synthesis flow for subthreshold SoC tapeout.

Flow Step	Modification	Conventional Approach
Cell Characterization	Characterize at sub $V_T$ $V_{DD}$	Synthesize at super $V_T$ and scale $V_{DD}$
Cell Characterization	Characterize at low temperatures	Characterize at high temperatures
Cell Characterization	Wide coverage of slew	Narrow coverage of slew
Cell Characterization	Cell pruning for OSV	Cell pruning for PPA metrics
Synthesis	Relaxed gate max delay constraint	No such constraint
Timing Closure	Conservative guardbanding	Guardband for jitter only



## 2.4 Results and Analysis

The resulting designed SoC (Figure 2.3) is the first wireless bio-signal processing chip enabling battery-free operation. The chip can be powered from an input as small as 30mV, enabling thermal energy harvesting. To the best of the authors' knowledge, this system has lower power, lower minimum input supply voltage, and more complete system integration than other reported wireless BSN SoCs to date. While the designed closed-loop power management 'stoplight' scheme [29] enables potentially indefinite operation while the node is worn, the flexible subthreshold datapath contributes significantly to reaching ultra low power levels that energy harvesters are capable of supplying. For example, in AFib detection experiment, the RR and AFib accelerators enable the transmitter (TX) and transmit the last 8 beats of raw ECG (buffered in the data memory) when a rare AFib event occurs. Measurement results for the AFib demo are presented in Figure 2.6. A pre-recorded set of AFib data from MIT-BIH database is used for this demo [30]. Detection occurs 12 RR intervals after the inception of an AFib event. A pattern recognition algorithm determines if an AFib has occurred [24]. The total chip power in the AFib experiment is 19 $\mu$ W, an ultra low power level partially enabled by the high energy efficiency processing capabilities of the digital subsystem, and the chip is powered exclusively from a 30mV harvested input.

### 2.4.1 Measured Results of Digital Processing Components

To quantify the energy efficiency the digital processing components exhibit, we performed speed and energy measurements on the individual accelerators and MCU GPP on chip. The 8-bit GPP MCU is a subthreshold RISC based on the PIC series [26]. The MCU is designed to run arbitrary programs and functions down to 0.26V, 1.2 kHz. Figure 2.7 shows the energy-delay (E-

D) curve for the MCU. The MCU consumes 0.7nW to 1.4 $\mu$ W measured power (0.26-0.55V) and 1.5pJ/op at the default 0.5V, 200 kHz setting.

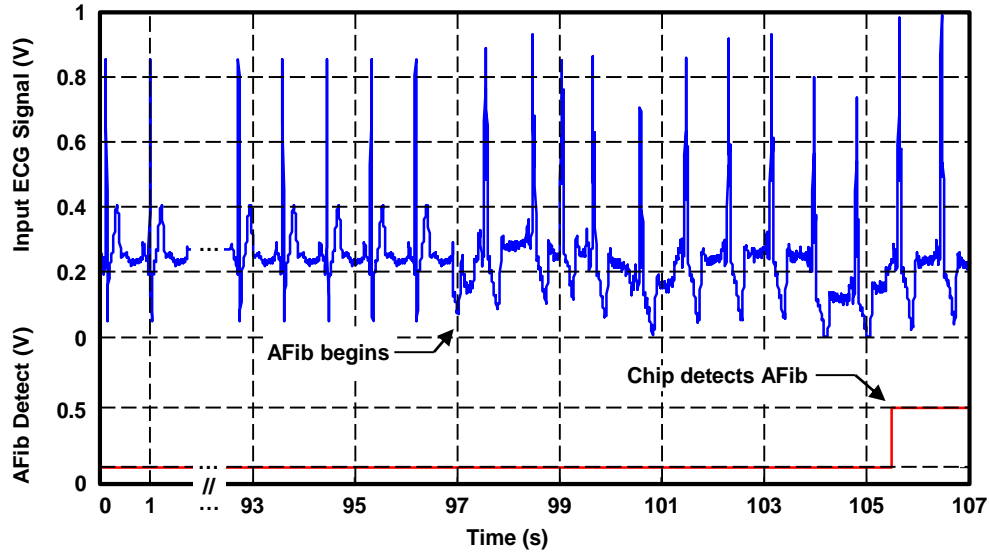


Figure 2.6. Measured system AFib demo experiment using RR extractor and AFib accelerator. Normal and atrial fibrillation heart waveforms from MIT-BIH database [30]. Total chip power in this mode is 19 $\mu$ W from a 30mV input [9].

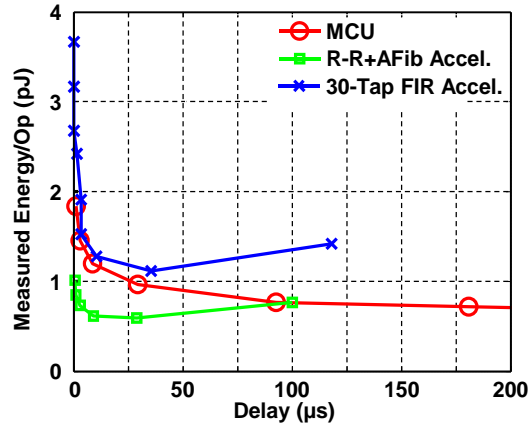


Figure 2.7. Measured energy-delay curves for MCU, RR+AFib, and 30-Tap, 1-Channel FIR [9].

A four-channel, programmable, max-30 tap, and synthesizable filter was designed to enable operation in the subthreshold regime down to 300mV (measured). The programmable options include coefficient selection, number of taps, and number of filters. The direct-form implementation of an FIR requires as many adders and multipliers as there are taps, costing area and leakage. Due to the small sampling rate for ExG signals, each result can instead be computed

serially over multiple faster clock cycles using only one multiplier and one adder. This architecture results in a 30x reduction in area per channel and a measured 1.1pJ per tap at 350mV [31]. The architecture saves valuable chip area and reduces leakage current. For further power reduction, each individual channel can be clock-gated. A measured energy-delay curve is given in Figure 2.7. The specific design with serial processing and programmability represent the customization advantage of choosing an ASIC platform.

The envelope detector circuit computes the average signal power within a specified frequency band. To reduce the computation complexity, division is implemented as simple right shifting. Further, the square operation is implemented as a lookup table. The envelope detector consumes 3.5nW (measured) at 0.5V and 200 kHz.

The heart-rate extractor accelerator is a simple version of the popular Pan-Tomkins algorithm [32]. This RR algorithm calculates the heart rate by means of time windowing and thresholding, after an initial 4 second time frame where the RR accelerator gains a baseline DC value for the heart waveform. The atrial fibrillation detector is an ASIC accelerator that detects the arrhythmia using an implementation of the clinically validated algorithm described in [24]. Many variables in the algorithm, such as the margin of error, are programmable. The algorithm uses a pattern recognition scheme that quantifies the entropy in 12 RR intervals. If the entropy is more than a programmable threshold, then an AFib event is reported.

Figure 2.8 (Left) presents a current breakdown of an RR extraction demo. In this in-vivo experiment, the chip ran an RR interval extraction algorithm and transmitted measured heart-rate every 5s operating from a 30mV supply voltage. The current is nearly evenly distributed among different components, and selective transmission significantly reduces the average power consumption of the transmitter. Current is used to represent power consumption in the pie chart

because different components and subsystems operate in different voltage domains. As such, it can be seen that the digital processing subsystem is a significant ( $\sim 33\%$ ) portion of the power. Thus, the importance of lowering the power and energy overhead of this subsystem through our proposed hardware selection and architecture design process is verified.

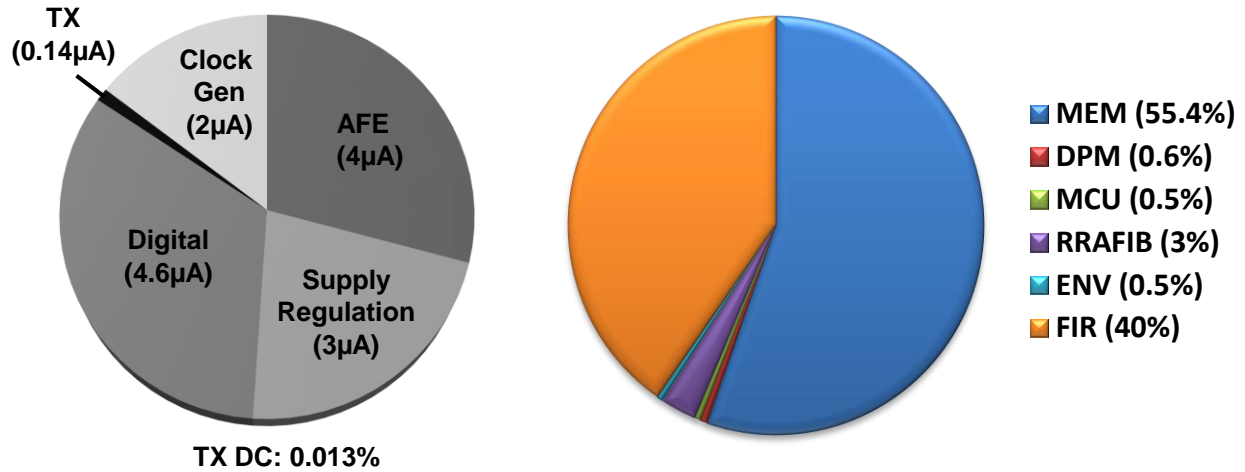


Figure 2.8. (Left) Current breakdown for RR extraction experiment. The digital processing subsystem contributes significantly to total drawn current. (Right) Relative energy per operation consumption of different components in digital subsystem, were they to each perform an operation during the same clock period. Memory power (mostly leakage power) is a huge contributor to digital subsystem power [25].

Figure 2.8 (Right) shows different components' within the digital processing subsystem's contribution to the total power and energy of the subsystem. Memory leakage dominates the subsystem's power consumption. Due to using ASIC accelerators, the bulk processing units consume minimal energy, aside from the FIR outlier. This is because the FIR can have 16-32 taps, which require multiple addition and multiply calculations per operation, making the FIR a heavy consumer of dynamic (switching) energy. It should be mentioned that the results for the MCU in Figure 2.8 and Figure 2.7 can be a bit misleading, as it would seem that the MCU consumes as much energy as ASIC accelerators. The clarification is that these numbers are on a *per operation* basis. In other words, the MCU consumes similar energy to one of the accelerators per clock period, but the MCU must take 100's or 1000's of clock cycles to complete the

calculations while an accelerator takes one, thus making the MCU much less efficient than an accelerator.

### **2.4.1 Results of Modified Synthesis Flow**

It was measured that the modified synthesis flow was a success, as correct module operation was tested across all 10 test chips supplied, and all chips passed the designed functionality tests.

Our coverage of slew degradation is successful, as across all corners characterized, the resulting slew from the worst case timing arc inputs was less than the amount of slew input during the characterization of that arc. For example, the greatest input slew characterized was 20FO4 slew and the greatest output load characterized was 16FO4 load. Under these conditions as inputs to the characterization of a size X1 inverter, the inverter outputs a slew of value 9FO4 slew, which is less than the 20FO4 slew input. What this means is that we have characterized beyond the worst case slew possible, which ensures correct and accurate timing information for the synthesis flow.

The relaxed gate max delay constraint, along with reasonable frequency constraints put on the modules' synthesis, results in designs with little duplicate logic paths, thus ensuring the correct synthesis results in the sense. We verified this by checking two items:

1. Maximum fanout/fanin (FO/FI) of the design both after synthesis and after place & route. A incorrectly synthesized design would have a smaller max FO/FI due to numerous duplicate logic paths, while a correctly synthesized design would have a larger max FO/FI. The range of max FO/FI for our designs was 13-18, a reasonable range.

2. The number of mapped register elements after synthesis, compared with the anticipated number of registers from RTL. For example, a 16b, 2 pipe-stage, 2 input, 1 output adder circuit should have  $16 \text{ (input A)} + 16 \text{ (input B)} + 17 \text{ (output Y)} = 49$  mapped registers. A correctly

synthesized design would have exactly 49 (or a number very close) registers, while a design with duplicate logic paths would have much more than the anticipated number. All of our designs have matching amounts of anticipated vs. actually mapped registers.

Our cell pruning methodology compelled us to rid the standard cell library of all size X1 inverters, NAND/AND2, and NOR/OR2 gates. Surprisingly, so long as a cell has more than 2 stacked transistors in both the pullup and pulldown paths (as in an AOI21 gate), they remain robust, even at size X1. This is because stacked transistors negate any parallel leakage paths (for example parallel NMOSs leaking in a NOR2 gate when both inputs are logic ‘0’), whose current draw may be close to or greater than the stacked pullup/down active current due to variation. With stacked transistors in the leakage path, it is much less likely that the leakage transistors’ current draw overcomes the active transistor’s current draw, thus leading to a more robust cell, even at smaller transistor sizes. Synthesis tools tend to map gates to their size X2 options to minimize dynamic and leakage energy due to the results of this pruning (Figure 2.9).

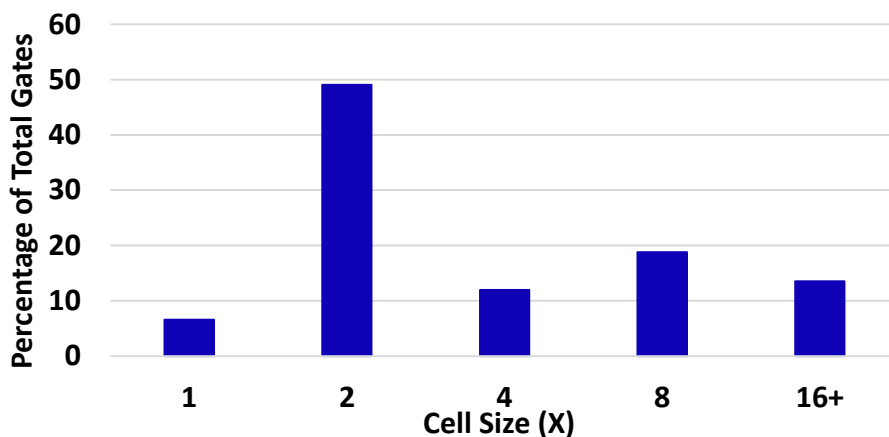


Figure 2.9. Average utilization of different cell sizes across 6 synthesized modules in the digital processing subsystem. Cell pruning takes away some size X1 cells, making synthesis tools favor size X2 cells to minimize dynamic and leakage energy. This trend does not change significantly among the 6 modules.

Lastly, a conservative guardband of 60ns and 200ns was given to hold and setup time respectively to robustly close timing. The 60ns of hold guardband was applied to all 5 checked

global corners, while the 200ns setup guardband was applied to the SS:0.45V:0°C slowest corner. Our post place & route netlist simulations show that hold buffers inserted contribute 19% of the total power and energy on average across 6 different modules. Though a conservative guardband was given to the setup time, no logic upsizing, logic restructuring, or setup optimization buffers were inserted during timing closure because the system clock frequency chosen for the chip (200kHz) is too slow to induce any setup timing issues, even with global and local variations apparent. We chose 200 kHz to be the system clock frequency due to the availability of an off-chip crystal.

## 2.5 Conclusions

Numerous innovative endeavors came together to accomplish the first wireless bio-signal processing chip enabling battery-free operation [9]. The highlights of those endeavors include low voltage boost converters, autonomous closed-loop power management, ultra low power amplifiers and transmitters, and subthreshold energy efficient on-chip digital processing. The contributions in this thesis regarding processing platform comparison and a modified synthesis flow for robust subthreshold digital design conceived the flexible and ultra low power processing datapath architecture, as well as its successful robust implementation. During demo experiments conducted on the chip, the digital subsystem consumes  $2.3\mu\text{W}$ , a number unachievable if not for our energy efficient processing architecture. Due to the extensive use of ASIC accelerators, processing of data is relatively ‘free’ in the sense of energy per operation. Unfortunately, memory leakage dominates and bottlenecks the digital subsystem’s energy consumption, so suppressing memory leakage would be a topic of importance for future revisions of the subsystem. The designs’ correct operation is ensured with a modified synthesis flow where robustness is the key constraint. However, many methods in the synthesis flow are crude and

conservative. Thus, they are overly pessimistic, which may incur great energy and area overheads. For example, hold buffers cost 19% extra power on average per module due to the conservative guardband placed during hold timing closure. We defer to the following Chapters 3, 4, and 5 for more refined methods. Nevertheless, the modified synthesis flow provides a quick and assuredly robust design flow.

The work in this chapter has been published in [YQ1] [YQ3] [YQ5] [YQ6] [YQ7] [YQ8] [YQ9] [YQ10].



# Chapter 3: Variation-Aware Path Timing

## Computation for Subthreshold Circuits

### 3.1 Background

Though sub $V_T$  circuitry provides attractive advantages in energy efficiency, the increased impact of PVT and on-chip variation (OCV) on gate delay presents a design challenge for timing closure at sub $V_T$  supply voltages. One of the major problems is timing closure in the face of on-chip variation. While the distribution of delay in superthreshold (super $V_T$ ) is Gaussian and the range of variation ( $\sigma/\mu$ ) across different logic paths is relatively small, the distribution in sub $V_T$  is log-normal and the range of  $\sigma/\mu$  is large. For example, the range of  $\sigma/\mu$  for delay across logic paths in a super $V_T$  ( $V_{DD}=1.1V$ ) synthesized design of an 8b PIC processor [26] at the TT:27°C global corner with low leakage devices in a 65nm technology is 6.7~12.2%, while the range for the same design in sub $V_T$  ( $V_{DD}=0.3V$ ) is 20.3~90.6% at the same corner (Figure 3.1). Looking at the critical path in super $V_T$ , because the distribution is Gaussian, the  $3\sigma$  yield (99.7%) delay is 659ps, which is only 9.1% more than the nominal delay of 604ps. However, in sub $V_T$ , the  $3\sigma$

---

This chapter is based off the published paper titled “Fast, Accurate Variation-Aware Path Timing Computation for Sub-threshold Circuits” [YQ12].

delay is 602ns due to log-normal distribution, or 126.4% more than the nominal delay of 266ns.

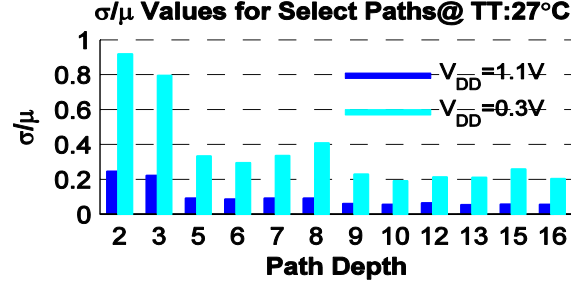


Figure 3.1. Comparison of  $\sigma/\mu$  values between super $V_T$  and sub $V_T$  regions. SubVT exhibits much more variation. The range of  $\sigma/\mu$  is much larger in sub $V_T$  across selected logic paths [33].

The greater than 100% increase in gate delay in this example due to stochastic delay clearly poses a major challenge for timing closure, and raises many questions.

First of all, unfortunately, traditional OCV timing closure methodologies are insufficient to deal with the acute sensitivity of stochastic delay in sub $V_T$  to  $\sigma/\mu$ , exacerbated by its wide range. Conventional timing closure methods employ a single  $\sigma/\mu$  for all potentially critical paths (the conventional derated timing method [7]). This method will not capture all the correct critical paths in a sub $V_T$  design. For example, in the previous example of the select logic paths in the synthesized PIC, the critical nominal delay path is 299ns with  $\sigma/\mu$  of 0.19, giving it a  $+3\sigma$  delay of 529ns according to log-normal distribution equations. However, the next longest path with nominal delay 266ns has  $\sigma/\mu$  of 0.23, giving it a  $+3\sigma$  delay of 530ns, making this the critical path. A monotonous derate value for all paths will incorrectly identify the critical path in sub $V_T$ . Therefore, this chapter proposes a fast tool to compute the derate value ( $\sigma/\mu$ ) for on-chip variation of delay for any logic path in a synthesized design for any given process corner.

The next issue is that we are uncertain whether or not conventional methods of calculating the derate values in super $V_T$  are still suitable for sub $V_T$ . For example, one of the prominent methods in super $V_T$  is known as the k-factor method [7][34], which deals with on-chip variations of temperature and supply voltage IR drop. However, the k-factor method assumes a linear

relationship between temperature (or IR drop) and delay. Given the change in the current equation and the mechanism of driving current in the sub $V_T$  regime, we highly speculate that this method is unusable in sub $V_T$ . Another method is to assign derate values to each individual standard cell, which is also subject to the logic path depth the cell resides in [35]. Not only does this give rise to massive look-up tables and prolonged cell characterization time, which impedes turnover time in design, but without understanding the statistical distributions in sub $V_T$ , as well as the accuracy of this method in sub $V_T$ , it would be dubious to employ the cell derate method. In summary, we propose re-verification of methods and/or conclusions that are established in super $V_T$  but may or may not still hold in sub $V_T$  due to the new design regime.

Lastly, there remains the issue of what factors determine the derate value in sub $V_T$ . Already starting to be acknowledged in super $V_T$  is the overly pessimistic flat derate values for robust timing closure [36][37]. This is highly undesirable in sub $V_T$  due to the already large  $\sigma/\mu$  values. An overly pessimistic estimation of  $\sigma/\mu$  would lead to an increasingly slow circuit and added area and energy penalty to cope with OCV. To remedy this, advanced OCV (AOCV) calculation methods have been proposed. AOCV looks to modify the derate values based on a variable factor that affects the  $\sigma/\mu$ , thus giving a more accurate derate value to perform timing closure with. For example, [34] gives a derate value based on expected temperature and supply voltage IR drop. [36][37] propose a tool that modifies the derate value based on logic path and cell location in layout. The problem arising here is that there is no systematic method that summarizes the multiple factors in determining the correct  $\sigma/\mu$  value for a logic path, nor is it clear which factors out of many are the main factors. Thus, in this chapter, we first argue that timing closure against voltage and temperature fluctuations should be solved through global corner characterization rather than modified  $\sigma/\mu$  values due to their drastic effect on delay variation in sub $V_T$ .

Afterwards, we propose a tool that has good accuracy in estimating  $\sigma/\mu$  for on-chip process variation at a given corner, which is compared against MC simulations. In this way, we provide a systematic approach to calculating the derate value on a per path basis, **thus addressing Challenge 4.**

## 3.2 Related Work

Much work has been done on the modeling of OCV, allowing designers to account for OCV before final Monte Carlo (MC) simulation and sign-off. In [38], the authors propose a highly accurate, computationally efficient method for predicting logic path delays at low  $V_{DDs}$  without need of MC simulation. Their approach is similar to ours in that they focus on the OCV at a given global corner. Also portrayed in this work is the advantage that their method is agnostic to what the specific current equation or what the delay distribution is. Their method is thus valid in the near-threshold region as well as subthreshold region, and is not bounded by complexity in determining the current equation or distribution of delay depending on what region of operation the transistors are in. However, their CAD integration flow requires multiple iterations of calculations, leading to a complex implementation flow. For each characterization of a standard cell library at each global corner, simulation time cannot be predetermined, as their method requires recursive simulation until the right ‘operating point’ is calculated. [39] provides a model of OCV that explores the effects of device size fluctuation and spatial correlation. Their work is conscious of a systematic approach of including global and local variations, as they split the calculation of each into two steps to determine the final gate delay. Their method pre-calculates the dependence of gate input load, output slew, and gate delay to input slew, output load, and device size variability in order to calculate the global, or inter-die variation. Their attention to the effects of slew on variation is unique. This correlates the  $\sigma/\mu$  of previous gates to later gates in

the logic path, which is a factor that conventional methods like per cell derate values [35] do not incorporate. However, when it comes to local, or intra-die variations, this work only considers spatial effects on  $\sigma/\mu$ , arguing that cells that are closer together in space are more correlated to each other's device size than those cells that are farther apart. Vital random components, such as random dopant fluctuations (RDF) and their effect on threshold voltage variation, are not captured. Also, the fact that intra-die variations must be calculated after layout is done and their spatial location determined means that the valuable information on variation is gained after physical implementation sign-off. While this work provides accurate delay estimates for timing checking, it would be cumbersome to use in doing timing optimization in the case that the chip fails timing closure, since cell locations would change after optimization and their stochastic delay values would have to be re-evaluated once again after the fact. Works such as [40][41][42] provide accurate, in-depth modeling of devices and estimation of variation in logic delay, but require deep understanding of device properties and physical mechanics. For example, in [40], an analytical treatment of slew's effect on OCV is given. The final analytical equation requires device parameters such as the standard deviation of the threshold voltage, and an  $\alpha$  value, which must be determined through deep understanding of the device operation, or found empirically. This becomes a drawback because the  $\alpha$  value is critical in determining the  $\sigma/\mu$ 's value with respect to gate input slew. Aside from this, this work offers no comment on how this method fits within the scope of path delay analysis and path timing closure, and this method is only verified on a single stage, per gate basis. [42] provides a  $\sigma/\mu$  estimation method that covers factors such as DIBL and spatial correlation. This work identifies and focuses on the main contributors to variation, which are gate length variation and random dopant fluctuation leading to threshold voltage variation. The model is proven accurate using a NAND2 gate as a verification vessel.

However, in order to use their final equation, the standard deviation of gate length and threshold voltage, as well as the relationship between gate delay, gate length, and threshold voltage must be pre-determined. The overhead of doing this is great and undesirable.

Aside from finding a systematic approach to correctly calculating the  $\sigma/\mu$  on a per path basis in sub $V_T$ , it is highly desirable to implement the method of calculation into the synthesis flow, another aspect that the related literature does not provide. After all, the end purpose of  $\sigma/\mu$  estimation methods is to predict the amount of OCV to aid in timing closure and timing optimization. To be able to integrate an estimation method into commercial timing closure flows that require timing checks at multiple corners, modes, and possibly multiple  $V_{DD}$ s conveniently, the method must be fast and easy to implement, applicable from early design stages, and scalable across  $V_{DD}$  and technology. We propose a method that requires extraction of only a few parameters that are easily obtainable through straight-forward simulation, is linear in implementation, requires little knowledge of the underlying process technology, and holds across a wide range of supply voltages, process corners, and technology nodes.

## **3.3 Solution**

### **3.3.1 Voltage and Temperature Variation in Subthreshold**

To understand the effects voltage and temperature variation have on delay in subthreshold, we ran MC simulations to find the path delay of a string of inverters at different voltages in subthreshold and different temperatures. Figure 3.2 gives the results of these simulations. There are drastic changes in delay and delay distribution depending on the voltage and temperature. Because of this wide spread in delay and delay distribution, it makes more sense to incorporate the effects of voltage and temperature through global corner cell characterization, instead of

adding them to the  $\sigma/\mu$  model. Thus, our model concentrates on the effects of process variation, and defer voltage and temperature variation to global corner cell characterization, thus covering across process, voltage, and temperature variations.

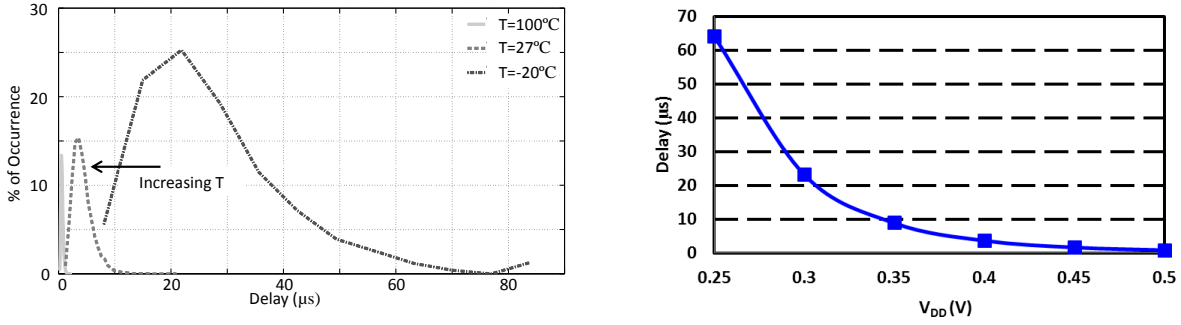


Figure 3.2. (Left) 10,000 point MC simulation results for delay of a string of inverters at 3 temperatures. Distribution drastically changes across different temperature. Increase of occurrences for  $T=-20^{\circ}\text{C}$  end tail (around  $80\mu\text{s}$ ) due to low MC sample number. (Right) Average delay of a string of inverters at different supply voltages. Delay varies drastically with  $V_{\text{DD}}$ .

### 3.3.2 Model Derivation in Superthreshold

Though the ultimate goal is to develop the method that is scalable across different supply voltages, we first describe our method in  $\text{super}V_T$ , where distributions are Gaussian and conclusions from prior works can be easily included. Modifications will be made when deriving the  $\text{sub}V_T$  method where the current equation changes.

#### 3.3.2.1 $\sigma/\mu$ of Single Stage Inverter

A natural starting point for the method's derivation is estimating  $\sigma/\mu$  for a single stage inverter. As many prior publications have re-iterated, the main cause [40][41] of OCV in deep sub-micron technologies is threshold variation. Our method will also assume this and will neglect secondary effects such as output load variation. Thus, the major factors of  $\sigma/\mu$  for an inverter are device sizing and input slew. To model the relationship between sizing and  $\sigma/\mu$ , we use the generalized results of [6] applied to delay as shown in eq. 3.1, where  $C_0$  is the process corner specific parameter to be extracted by simulation, and  $s$  is the normalized gate size.  $C_0$

should be extracted under ideal slew conditions, to isolate the effect of device sizing and input slew. To model the effects of slew, we observe that the results of [40] can be easily generalized with a 2 segment piece-wise linear function that will portray the relationship between input slew and  $\sigma/\mu$  as in eq. 3.3, where  $a_1, a_2, b_1, b_2$  (since there are two segments of the piece-wise linear line there are two sets of  $a$  and  $b$ ) are extracted parameters at a set process corner,  $x$  is a normalized slew number, and  $\mu$  is the mean delay based on load/drive ratio. When slew is small,  $a_1$  and  $b_1$  are used, and  $a_2$  and  $b_2$  are used when slew is large. The boundary condition for ‘small’ and ‘large’ skew is when  $x=(b_2- b_1)/(a_1- a_2)$ , the point at which the two line segments cross. Since these factors are acting on top of each other, the total  $\sigma/\mu$  is the sum of eq. 3.1 and eq. 3.2 as in eq. 3.3.

$$\sigma_{1stage}^2(device\ sizing) = \mu^2(C_0/s) \quad \text{eq. 3.1}$$

$$\sigma_{1stage}^2(input\ slew) = \mu^2(a_{1(2)}x + b_{1(2)}) \quad \text{eq. 3.2}$$

$$\sigma_{1stage}^2 = \mu^2(C_0/s + a_{1(2)}x + b_{1(2)}) \quad \text{eq. 3.3}$$

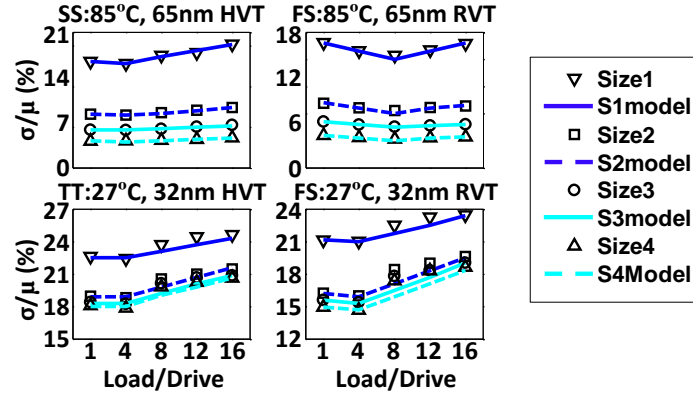


Figure 3.3. Model vs. MC simulation of single stage inverter pulldown  $\sigma/\mu$  across different process corners sweeping device size and load/drive ratio (input slew). Results show close matching between model and MC simulation results.  $V_{DD}=0.9V$  [33].

To extract the necessary parameters, simple MC simulations on an inverter chain varying device size and slew (which can be accomplished by varying the output load of each stage) are done (we will refer to this as *simulation 1*, depicted in Figure 3.4). In our experiment, we ran MC



simulations on a 32 stage inverter chain, where the device sizes were varied, and slews were set by changing the load/drive ratio. The results of the simulations were matched against the model (Figure 3.3) across various corner, threshold voltage ( $V_T$ ) options, and technology nodes. A separate set of parameters should be extracted for N- and P-type devices.

### 3.3.2.2 Expansion of $\sigma/\mu$ to Path Delay

In this section we expand the  $\sigma/\mu$  calculation to logic paths. Our difficulty in doing this is that the logic stage to stage delay variations are not independent of each other. This is because the input slew to each stage varies somewhat since the driving current of the previous stage is also subject to variation. Thus, the  $\sigma/\mu$  of the later logic stage is correlated to the  $\sigma/\mu$  of the previous stage. The effect of slew variation must be included in path delay  $\sigma/\mu$  calculation or we will have underestimated the  $\sigma/\mu$  value.

Since the path delay  $\mu$  is the sum of individual  $\mu$ 's of each stage, it would be convenient to model the slew variation effect with a coefficient of correlation, parameter  $\rho_1$ , which is simply the correlation factor between  $\sigma/\mu$  for adjacent logic stages.  $\rho_1$  can easily be extracted from the results of *simulation 1*.  $\rho_1$  values across device sizing, output slews, process corner, threshold voltage options, and technology nodes are shown in Figure 3.5 for pulldown driving pullup transitions.

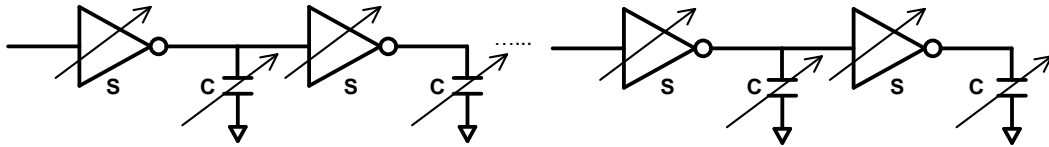


Figure 3.4. Diagram of circuit for *simulation 1*, where a chain of inverters' device size ( $s$ ) and load/drive ratio ( $c$ ) are swept uniformly across all stages in the inverter chain.

The  $\rho_1$  values are obtained by taking the average of all correlation coefficients in the inverter chain between two adjacent logic stages. The  $\rho_1$  value can be viewed as constant for a given process corner. As output slew and device size were swept,  $\rho_1$  did not vary more than 0.05 for

any corner (Figure 3.5). This leads to great convenience as now a logic path's  $\sigma$  can be calculated as in eq. 3.4, where  $i$  represents a logic stage.

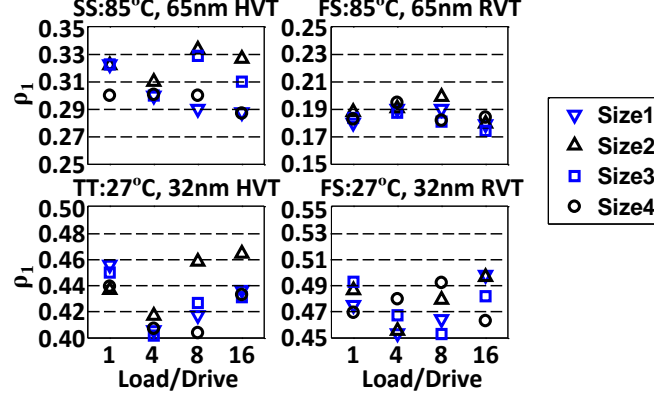


Figure 3.5. Extracted  $\rho_1$  values from MC simulation on inverter chain pulldown transitions across different operating conditions.  $\rho_1$  displays relatively constant value for a given corner [33].

$$\begin{aligned}\sigma_{path}^2 &= \sum \sigma_i^2 + \sum Cov(i, i+1) \\ &= \sum \sigma_i^2 + 2\rho_1 \sum \sigma_i \sigma_{i+1}\end{aligned}\quad \text{eq. 3.4}$$

A different  $\rho_1$  value should be extracted for pullup transitions driving pulldown transitions, especially for FS and SF corners where the values can be quite different. Thus, eq. 3.4 can be modified to eq. 3.5, where  $\rho_{1N}$  denotes the coefficient for pulldown driving pullup transitions,  $\rho_{1P}$  denotes the coefficient for vice versa, and  $n$  denotes the number of logic stages in the path.

$$\sigma_{path}^2 = \sum_{i=0}^{n-1} \sigma_i^2 + 2\rho_{1N} \sum_{i=0}^{\frac{n}{2}-1} \sigma_{2i} \sigma_{2i+1} + 2\rho_{1P} \sum_{i=0}^{\frac{n}{2}-1} \sigma_{2i+1} \sigma_{2i+2} \quad \text{eq. 3.5}$$

Equation 3.5 means we can easily calculate a logic path's  $\sigma/\mu$  value. The individual  $\sigma$  values of each stage can be attained using eqs. 2-4 for inverters, and the results of the following Section 3.3.2.3 for any generic logic gate. This model assumes that  $\rho_1$  is constant, which may not always be true, but we will show empirically that this assumption results in good accuracy for  $\sigma/\mu$  values for a broad range of conditions (Figure 3.8, Figure 3.10, Figure 3.11, and Figure 3.12).

### 3.3.2.3 Expansion of $\sigma/\mu$ to Generic Logic Gates

This section expands the  $\sigma/\mu$  calculation to any logic gate. We can easily estimate a gate delay's  $\mu$  value from logical effort and a slew and load vs. delay lookup table. The lookup table can easily be obtained by running a nominal simulation of an inverter chain and sweeping the input slew and output load, and measuring delay. We will refer to this as *simulation 2*. Though gate delay will fluctuate somewhat based on input patterns, the fluctuation has little effect on the final  $\sigma/\mu$  estimation, or adjustments for different input patterns can be made for better accuracy.

To estimate  $\sigma$  of delay for any given logic gate, we propose the following approximation. Without loss of generality, we may interpret stacked transistors as multiple stages of pass transistor gates that only either pullup or pulldown. Hence, a stacked pullup/pulldown can be interpreted as a 'mini logic path' composed of only pass gates. Thus, the model for  $\sigma/\mu$  for any generic logic gate has already been derived: each transistor's delay  $\sigma$  would follow eqs. 3.1-3, and the total  $\sigma$  would follow eq. 3.4. For simplicity, we assume that the delay during a transition through each transistor (pass gate) in the stack is equal regardless of the individual transistor's position in the stack in order to use eqs. 3.1-3. In fact, the single stage inverter model is merely a special case of this approximation where there is only one pass gate in the 'mini logic path'.

We propose extracting a different coefficient of correlation value  $\rho_2$  to model the correlation between transistors in a stack. This is because, unlike logic stage to stage correlation due to slew propagation, transistors in a stack mostly affect each other through body-effect modulation of their threshold voltages. Thus, the correlation mechanism is different, and a different  $\rho_2$  should be extracted. Also, similar to  $\rho_1$  extraction, a different  $\rho_2$  value should be extracted for pullup transitions ( $\rho_{2P}$ ) and pulldown transitions ( $\rho_{2N}$ ). Finally,  $\rho_2$  should replace  $\rho_1$  in eq. 3.4, and any generic logic gate's  $\sigma/\mu$  can be estimated. To summarize and clarify the model for single gate

$\sigma/\mu$ , eqs. 3.3-4 are re-written as eq. 3.6-7, where the new parameter  $N$  is the stack depth, and  $i$  represents a transistor in the stack.

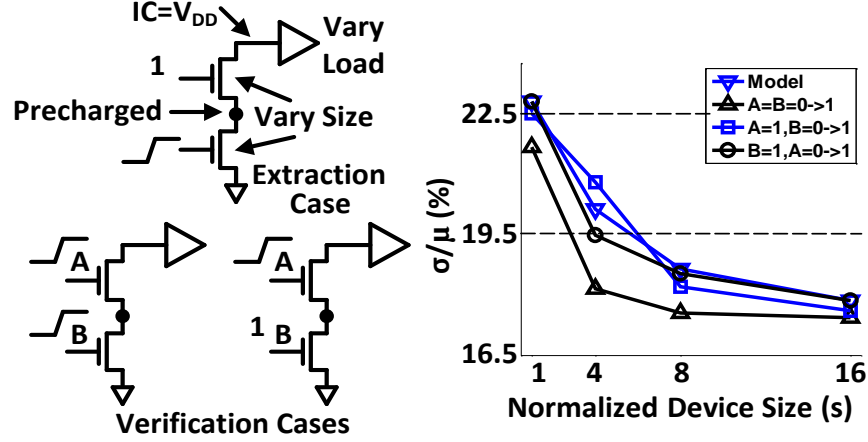


Figure 3.6. (Left) Example simulation setup to extract and verify  $\rho_2$ . MC simulations are done sweeping transistor size, load, and stack depth to extract delay through the pulldown/pullup transistor. (Right) Model vs. MC simulation results for NAND2 pulldown  $\sigma/\mu$ . Results show close matching [33].

$$\sigma_{1\text{ transistor}}^2 = (\mu/N)^2 (C_0/s + a_{1(2)}x + b_{1(2)}) \quad \text{eq. 3.6}$$

$$\sigma_{\text{logic gate}}^2 = \sum \sigma_i^2 + 2\rho_2 \sum \sigma_i \sigma_{i+1} \quad \text{eq. 3.7}$$

The parameter  $\rho_2$  can be extracted running MC simulations on stacked transistors, sweeping device sizes, output load, and stack depth. We ran 1000 point MC simulations per configuration (device size, output load, and stack depth) to extract  $\rho_2$ . Special attention should be observed on the input pattern so that all internal nodes are pre(dis)charged, since  $\rho_2$  is modeling the correlation of delay between internal nodes of the stack. If not all internal nodes are pre(dis)charged, some nodes will not swing to  $1/2V_{DD}$ , and there will be no way to extract  $\rho_2$  from the simulation. Instead, simulations of these other input patterns provide an extra layer of verification. In our experiment, we measured the entire stack's  $\sigma/\mu$ , and compared it to the modeled  $\sigma/\mu$  from eq. 3.7. The results with good accuracy (Figure 3.6 (Right)) verify that regardless of the input pattern our approximation of a stack viewed as a pass gate ‘mini logic

path' holds well. We will refer to this as *simulation 3*, and its setup is depicted in Figure 3.6 (Left), using 2-stack NMOSs as an example. We also simulated 2, 3, and 4 stacks. Through these simulations, we found that  $\rho_2$  has a relatively constant value for a given corner across device size, output load, and stack depth.

### 3.3.2.4 $\sigma/\mu$ Estimation Method for Superthreshold

Combining the results from the prior sections, we have an approach to estimate  $\sigma$ ,  $\mu$ , and the  $\sigma/\mu$  value for any logic gate and any logic path without need for extensive MC simulation for each logic path. Our methodology is summarized in Figure 3.7.

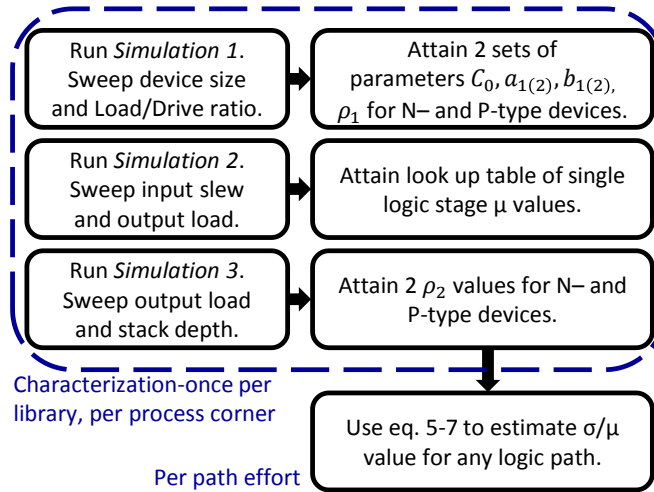


Figure 3.7. Summary of  $\sigma/\mu$  estimation method. Our method is linear in implementation and has high level of abstraction [33].

A few observations can be made about our method. First, the implementation is strictly linear. *Simulations 1-3* are run once in a characterization step for a given standard cell library and process corner, the relevant parameters are extracted from the simulations, and subsequent computations of  $\sigma/\mu$  for each path are guaranteed to complete. No loops or iterative processes are needed, making the implementation simple. What's more, all the parameters extracted come from measurements of delay. For example,  $\rho_1$  is the coefficient of correlation between delays, and not between any underlying device parameters. This means that the method has a high level

of abstraction, making the method easy to understand and scale from technology to technology. Finally, since the circuits simulated are ‘basic constructs’ like inverter chains and charge/discharge stacks, the method can be completed and  $\sigma/\mu$  known very early in the design process. We applied this method to a synthesized PIC processor [26] in two technology nodes across various process corners. The modeling of several select logic paths after completion of place and route using the proposed method is presented in Figure 3.8 and verified against MC simulation of those paths. For all logic paths, we assumed setup, hold, and clock-to-q times to be constant. Future work will involve expanding our method to include estimation of stochastic setup and hold time. The paths include gates such as MUX, XOR2, XOR3, and ADDF (full adder, mirror adder topology) cells, thus containing a wide span of standard cells.

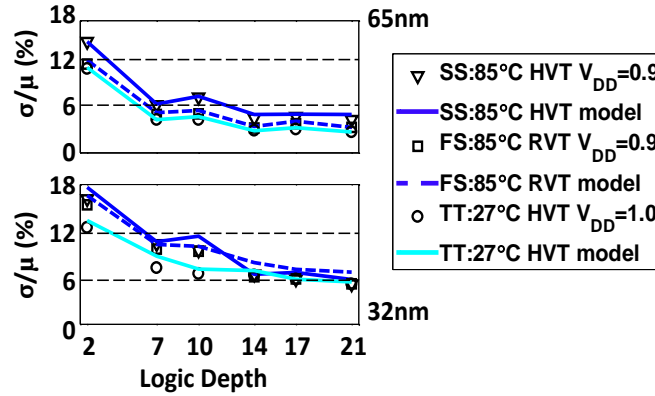


Figure 3.8. Model vs. MC simulation of selected paths’  $\sigma/\mu$  in synthesized processor. Results show close matching [33].

### 3.3.3 Model in Subthreshold

The MOSFET current equation changes in the sub-threshold region, which means we must adjust or validate the proposed method to ensure its applicability to low supply voltages.

#### 3.3.3.1 Device Size in Subthreshold

The generalized results of [6] can be applied to delay because the relationship between drive current and threshold voltage is roughly linear in strong inversion. Thus, as  $\sigma_{V_T}^2 \propto 1/s$ , so is

$\sigma_{ID}^2 \propto 1/s$ . However, current is exponentially dependent on threshold voltage in sub $V_T$ . We can derive the relationship between device sizing and variability in current (and in turn delay) starting from the subthreshold current equation (eq. 3.8), where  $I_o$  is the current when  $V_{GS}=V_T$ ,  $n$  is a constant equal to the ratio between capacitance of the depletion layer and oxide layer plus one ( $n=1+C_D/C_{OX}$ ),  $V_{th}$  is the thermal voltage,  $K_2$  is a constant that reflects the conclusions of [6], and  $s$  is the normalized device size.

$$I_D \cong I_o \frac{w}{L} \exp((V_{GS} - V_T)/nV_{th}) \quad \text{eq. 3.8}$$

$$Var[I_D] = K_1^2 Var[\exp(-\frac{V_T}{nV_{th}})], K_1 = I_o \frac{w}{L} \exp(\frac{V_{GS}}{nV_{th}})$$

let  $\mu_o, \sigma_o$  be mean, sigma of  $V_T$  variation.  $\sigma_o^2 = K_2/s$

$$Var[I_D] = K_1^2 (\exp(\sigma'^2) - 1) * \exp(2\mu' + \sigma'^2), \quad \text{eq. 3.9}$$

$$\mu' = -\frac{1}{nV_{th}} \mu_o, \sigma'^2 = (\frac{1}{nV_{th}})^2 \sigma_o^2$$

$$Var[I_D] = K_1 (\exp(\sigma'^2) - 1) * E[I_D]^2 \quad \text{eq. 3.10}$$

$$Var[I_D] \cong E[I_D]^2 K_1 \exp(\sigma'^2), \text{ let } K_3 = K_2 (\frac{1}{nV_{th}})^2 \quad \text{eq. 3.11}$$

$$Var[I_D] \cong E[I_D]^2 K_1 \exp(\frac{K_3}{s})$$

$$\sigma_{1stage}^2(device\ sizing) = \mu^2 C_1 \exp(\frac{C_2}{s}) \quad \text{eq. 3.1(b)}$$

From the derivation we conclude that eq. 3.1 must be modified to eq. 3.1(b) for use in sub $V_T$ , where  $C_1$  and  $C_2$  are constants extracted from *simulation 1*. A few important notes are made about the derivation. Properties of log-normal distributions are used to arrive at eq. 3.9 and 3.10, where the term  $-\frac{V_T}{nV_{th}}$  is a Gaussian variable. We assume that  $\exp(\sigma'^2)$  is large such that  $\exp(\sigma'^2) - 1 \cong \exp(\sigma'^2)$  to arrive at eq. 3.11 from eq. 3.10. Our conclusion about device size

and variability in  $\text{sub}V_T$  is verified through matching with MC simulation (*simulation 1* in  $\text{sub}V_T$ ). The results are presented in Figure 3.9.

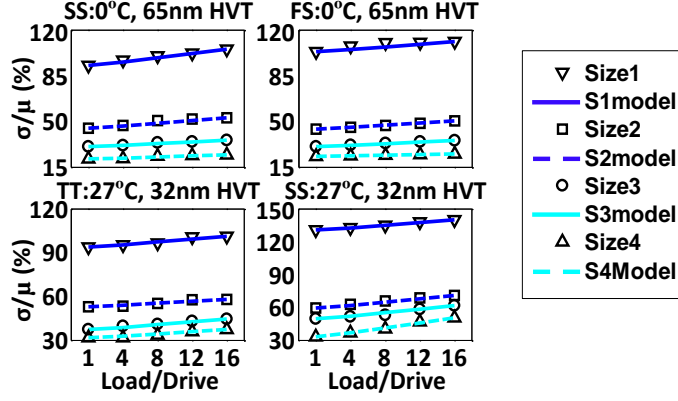


Figure 3.9. Model vs. MC simulation of single stage inverter pulldown  $\sigma/\mu$ . Device size and input slew (load/drive ratio) are swept. Supply voltage set at  $\sim 100\text{mV}$  below threshold [33].

### 3.3.3.2 Slew Dependence in Sub-threshold

The relationship between slew and delay variability for the sub-threshold region can be derived similarly to [40] for the superthreshold region. The derivation begins with solving the differential equation for  $I_D$ . The derivation assumes a linear ramp for the input slew ( $T_{slew}$ ), as in eq. 3.12. Also, for convenience, the time  $t$  at which rail to rail swing is accomplished is defined as the delay  $t_{delay}$  in this derivation, as in eq. 3.13.

$$I_D \cong I_o \frac{w}{L} \exp((V_{GS} - V_T)/nV_{th})$$

$$V_{GS} = \frac{t}{T_{slew}} V_{DD} \quad \text{eq. 3.12}$$

$$I_D \cong K_4 \exp(\frac{t}{T_{slew}} V_{DD} - V_T)/nV_{th}), K_4 = I_o \frac{w}{L}$$

$$\text{solving } C_L = \frac{dV_{out}}{dt} \cong I_D \text{ gives}$$

$$V_{out} = K_4 K_5 \exp(\frac{t}{T_{slew}} V_{DD} - V_T)/nV_{th}), K_5 = \frac{T_{slew}}{V_{DD} n V_{th} C_L}$$

$$\text{solve for } t \text{ and let } V_{out} = V_{DD}$$

$$t_{delay} = \frac{T_{slew}}{V_{DD}} (nV_{th} \ln \frac{V_{DD}}{K_4 K_5} + V_T) \quad \text{eq. 3.13}$$



$$Var[t_{delay}] = \frac{T_{slew}^2}{V_{DD}^2} Var[V_T]$$

$$thus \sigma_{t_{delay}}^2 \propto T_{slew}^2, or \sigma_{t_{delay}} \propto T_{slew}$$

The derivation leads us to conclude that  $\sigma \propto T_{slew}$ . This means eq. 3.3 is applicable to  $subV_T$ , where only one set of  $a_1$  and  $b_1$  is needed. This conclusion is verified in Figure 3.9.

### 3.3.3.3 Logic Paths and Gates in Subthreshold

Reference [43] proposes that the sum of log-normal random variables is closely log-normal, which we apply to extend our calculation of  $\sigma/\mu$  in  $subV_T$  to logic paths and gates. Since the mechanisms of slew variability and body bias modulation of threshold voltage still apply in  $subV_T$ , eqs. 3.5-7 are applicable to  $subV_T$  as well. When calculating stochastic delay in  $subV_T$ , we can assume the path's delay follows a log-normal distribution. It should be noted that after running *simulation 2* to get single logic stage delay  $\mu$  values, subthreshold logical effort [44] should be used in place of conventional superthreshold logical effort analysis. Estimations based on this approximation are presented in Figure 3.10 and Figure 3.11.

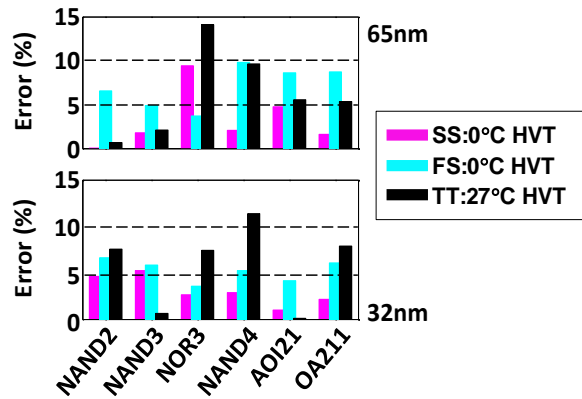


Figure 3.10. Model vs. MC simulation error of single stage logic gates'  $\sigma/\mu$  in sub-threshold region. Results show good accuracy. Results are for a timing arc with stacked transistors (e. g. pullup path for NOR3 gate). Pulldown stack results presented for AOI21 and OA211 gate. Supply voltage set at  $\sim 100\text{mV}$  below threshold [33].

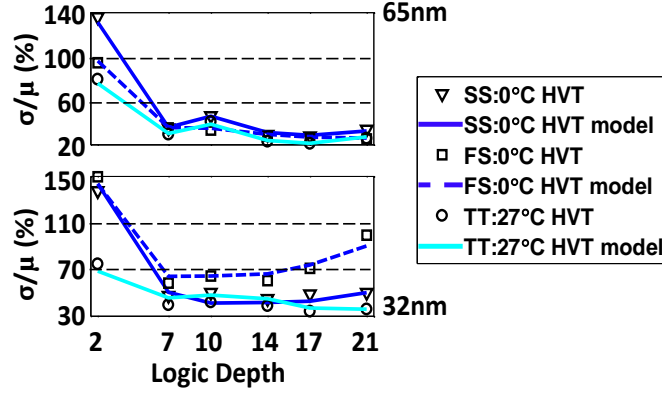


Figure 3.11. Model vs. MC simulation of selected paths'  $\sigma/\mu$  in subthreshold [33].

#### 3.3.3.4 Summary of Method in Subthreshold

The previous sections conclude that the methodology derived for  $\text{super}V_T$  is applicable to  $\text{sub}V_T$  as well with a few modifications, thus ensuring that our method is scalable across a wide range of supply voltages. Figure 3.11 shows results for selected logic paths in a synthesized PIC processor in  $\text{sub}V_T$ .

## 3.4 Results and Analysis

Figure 3.3 shows results for our model of a single stage inverter in  $\text{super}V_T$  (eq. 3.3). It displays good matching for our interpretation of device size and slew's effect on variation. The error for eq. 3.3 is  $<2\%$  from the MC simulations we ran. This very high accuracy is important as it lays the foundation for all the rest of the model's derivation. Figure 3.5 shows results that  $\rho_1$  values remain relatively constant given a process corner. A key observation we make during this experiment is that correlation only occurs between logic stages that are directly adjacent to each other in the path. In all our simulations, the coefficient dropped to near zero for all cases that did not meet the adjacent criteria. Using the results of Figure 3.3 and Figure 3.5, any  $\sigma/\mu$  can be calculated with very high accuracy for a string of inverters. In fact, we hypothesize that the lower

accuracy our final results exhibit are due to imperfections on assumptions made on logic gates with stacked transistors.

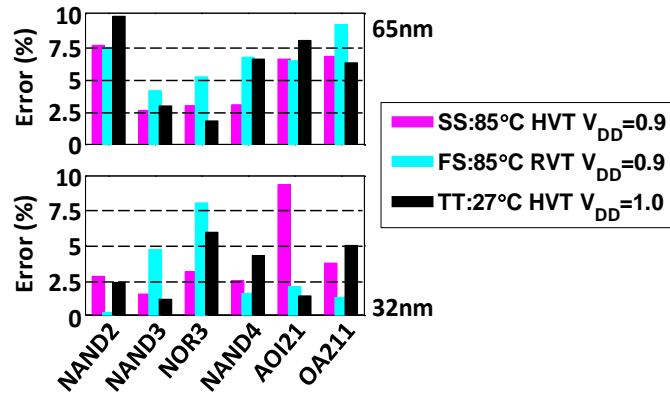


Figure 3.12. Model vs. MC simulation error of single stage logic gates'  $\sigma/\mu$ . Results show good accuracy. Same arc used as in Figure 3.10 [33].

Figure 3.8 displays the results based on MC simulation when we expand the model to all gates. Figure 3.12 displays the results for full path  $\sigma/\mu$  calculation. The amount of error when using this model for single stage logic gates are presented in Figure 3.12, which shows <10% error. The error in  $3\sigma$  yield stochastic path delay is 8.5% (Figure 3.8). As can be seen, the accuracy deteriorates (though it is still good) when we expand the model to incorporate any logic gate. We speculate that there could be several reasons for this. First, our previous assumption that each transistor in a pullup/pulldown stack contributes evenly to the total delay through the stack is clearly not accurate. Through Spice simulations as well as circuit analysis we know that all transistors except the one(s) closest to the gate's output node (dis)charge their drain node much faster. This fact may contribute greatly to the lower accuracy. We speculate that weighting the transistors according to their contribution to total stack pulldown/pullup delay will improve accuracy. Next, we didn't incorporate the slight changes in  $\mu$ , and possibly  $\sigma$ , that different input patterns to the gate presents. Finally, we made no assumptions (or in other words assumed transistors' variation to be independent) about transistors that pulled down/up in parallel.

Another place that we think induces more inaccuracy is the assumption of constant  $\rho_1$ . When we ran MC simulations to verify our model for full logic paths, we noticed that  $\rho_1$  exhibited lower values for gates that drove large fanouts ( $>32$ ).

Figure 3.9 shows that our model for inverters is still very accurate in  $\text{sub}V_T$ . The error here is  $<3\%$ . We can see that as predicted, variation linear increases with slew in  $\text{sub}V_T$ . The same issues for decreased accuracy persist in  $\text{sub}V_T$ , as can be seen in Figure 3.10 and Figure 3.11. The error when estimating gate  $\sigma/\mu$  is  $<15\%$ , and error is 10.6% for  $3\sigma$  yield stochastic path delay using log-normal distribution calculations.

### 3.4.1 Correct Critical Path Identification

The proposed estimation method provides a useful tool that can aid in identifying the correct critical paths in a  $\text{sub}V_T$  design. To understand this, we notice that in the  $\text{super}V_T$  case, the range that  $\sigma/\mu$  values span for potentially critical paths (logic depth is 14-21 for synthesized PIC) is very small. For example, in the 65nm technology at  $TT:27^\circ\text{C}$ , the  $\sigma/\mu$  values for potentially critical paths has a range of 0.84%. Thus, a single  $\sigma/\mu$  value can be used to model the OCV on all of these paths (as is often done in conventional practice). The range of  $\sigma/\mu$  values for a  $\text{sub}V_T$  design, though, is large. For example, in the 65nm technology at  $TT:27^\circ\text{C}$ , the range is 20.52%. It is possible then, that in  $\text{sub}V_T$ , a path that has shorter nominal delay with a greater  $\sigma/\mu$  value may have a greater  $+3\sigma$  yield delay than a path with greater nominal delay but smaller  $\sigma/\mu$  value. The path with shorter nominal delay, then, becomes critical. Since the proposed estimation method has good accuracy to track the  $\sigma/\mu$  value on a per path basis, the advantages of easy obtainability and quick turnaround are emphasized within this context.

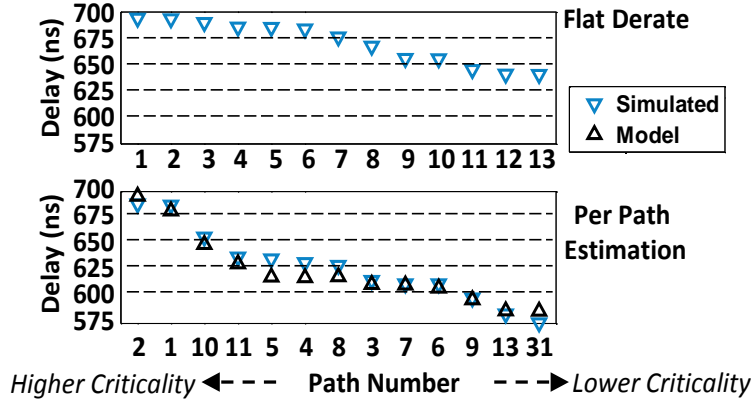


Figure 3.13. Critical path delays before and after using the proposed method to estimate  $\sigma/\mu$  values and stochastic path delay. Results are for  $3\sigma$  yield delay. Figure shows different path priorities based on timing closure method used [33].

The proposed method's advantages are exemplified when we applied it to timing closure on a sub $V_T$  FIR. The method was integrated into a standard synthesis and place and route flow using custom scripts. Two closure runs were done for the 16b, 16tap FIR for comparison. The first used a conventional flat timing derate on all potentially critical paths (the first 7% of paths reported by the timing closure tool), and the second applied our proposed estimation method on a per path basis. Figure 3.13 shows the results of the two runs under a  $3\sigma$  yield constraint at TT:27°C for 65nm. The paths are numbered according to their critical priority during the flat derate closure run. As we can see, using the proposed method to estimate the  $+3\sigma$  delay changes the order of critical priority. For example, in the case of Path 10, we see that its actual priority is 3. Thus, it is especially important to be aware of the wide range of  $\sigma/\mu$  values in sub $V_T$ , and the proposed method is an efficient way to track the paths' true priority when performing ECO (engineering change order) flows for timing closure in sub $V_T$ , in addition to efficiently estimating the paths'  $\sigma/\mu$  value. The values shown in Figure 3.13 are verified through matching with MC simulation.

## 3.5 Conclusions

The simplicity, high level of abstraction, and straight forward implementation of our estimation method makes it much easier to integrate into the design flow to provide per path timing closure than other per path modeling efforts (e. g., [38][39]). For example, the model proposed in [39] becomes computationally complex since the  $\sigma$  values for each gate must be pre-computed based on numerous sensitivities. Similarly in [38], though the computation is more efficient, entire standard cell libraries must be characterized on a per gate basis. Our method which is based on extraction of a few easily visible parameters from basic circuit constructs, which in turn can cover estimation of variability for any generic gate and logic path, thus leads to faster turnaround and easy integration. However, the drawback is that this leads to less accurate results. For example, [38] accomplishes 5% error compared to our  $<11\%$ .

We have proposed an estimation method for attaining the  $\sigma/\mu$  value of delay for logic paths in synthesized designs. The method is easily obtainable through extraction of several parameters from a few simple characterization simulations that can be performed once per library. The method requires little understanding of the underlying physics of devices for the designer and provides the designer with a good picture of the effect of OCV beginning from early design stages. The method is usable in the sub-threshold region, and its efficient use in this region is exemplified by providing timing closure flows with each logic paths' true critical priority when the method is integrated into standard flows with custom scripts. The resulting method shows good accuracy, with error of 10.6% when estimating sub $V_T$  path delay for all paths we simulated.

To remedy the drawback of decreased accuracy, further research should be done by closely studying the factors that contribute to a generic logic gate's  $\sigma/\mu$ , as well as outliers in  $\rho_1$  values.

Future work will also involve expanding our method to include estimation of stochastic setup and hold time.

The work in this chapter has been submitted for publication [YQ12].

# **Chapter 4: Timing Closure for Subthreshold Circuits Using a Two-Phase, Latch Based Timing Method**

## **4.1 Background**

Supply voltage scaling into the subthreshold region is becoming an increasingly attractive solution to save energy and power in cases where performance is not the driving factor, for example in body sensor nodes (BSNs) [9][19]. However, the increased impact of PVT variation on gate delay presents a design challenge for timing closure at  $\text{sub}V_T$   $V_{DD}$ s. Our Monte Carlo (MC) simulations on a size X1 inverter from a commercial 65nm technology show FO4 delay changing 12X from SS:27°C corner to FF:27°C corner at  $V_{DD}=0.3V$ , which is  $\sim 130mV$  below  $V_T$

---

This chapter is based off the published paper titled “Hold Time Closure for Subthreshold Circuits Using a Two-Phase, Latch Based Timing Method” [YQ11].



(Figure 1.1, redrawn here). Delay distributions due to mismatch display log-normal distributions, which have greater  $\sigma/\mu$  values than the Gaussian distributions that result from mismatch in superthreshold. Conventional synthesis methodologies are not well designed to cope with this extreme spread in delay, often resulting in greatly increased logic area to meet setup time and excessive hold buffer insertion to fix hold time. This is especially disconcerting in sub $V_T$  as the added area leads to increased leakage and dynamic energy, possibly undermining our original goal of ultra low power operation.

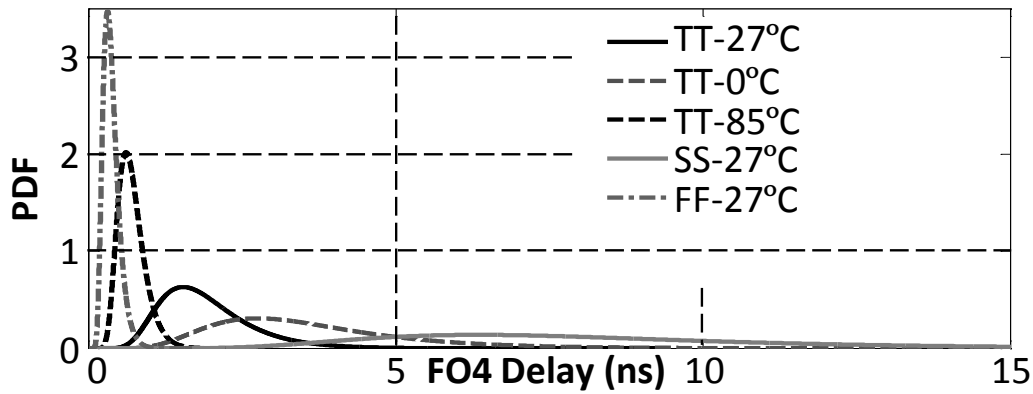


Figure 4.1. Distribution of delay of size X1 inverter in 65nm process across different process corners. Distributions display log-normal tendencies with long end tail and wide spread [45].

In addition, though there is work proposing fixes to setup time failure [46], little work has been done to study hold time closure in sub $V_T$ . This chapter will show through analysis and simulation how hold failures occur and how conventional buffer insertion will lead to great energy overhead. Thus, we propose an alternative two-phase, latch based timing method solution that eliminates hold buffers entirely for robust, low power hold time closure, **addressing Challenge 3**. We show that compared to conventional hold buffering, our solution saves up to 37% (at  $6\sigma$  yield) in energy per operation and allows for post tapeout hold time correction. Replacing registers with latches also permits time borrowing, which we show can save up to 47% in total energy per operation ( $6\sigma$  yield) when used for setup time closure.

### 4.1.1 Conventional Timing Closure

Current conventional timing closure flows ensure circuit robustness by setting reasonable guardbands, often pessimistic derates for OCV, and thorough timing checking at all the correct global corners. Current flows for timing closure include steps such as:

1. Performing timing checking at all global corners
2. Timing checking at extreme global corners (for example in sub $V_T$ , SS:0°C, FF:45°C-FF)

with flat guardband for both setup/hold to account for effects of clock jitter.

3. Timing Checking at extreme global corners with derated timing metrics for gates to ensure  $6\sigma$  yield.

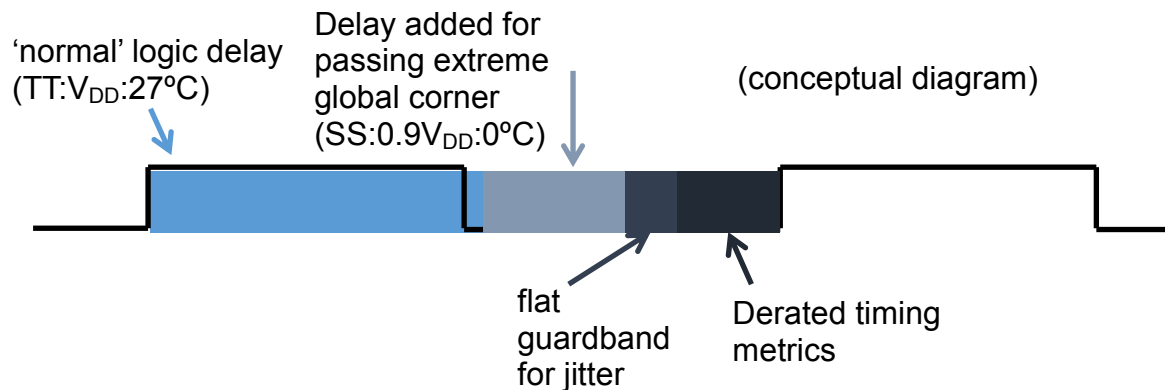


Figure 4.2. Conceptual diagram on how performance degrades due to various sources of variation. In an effort to minimize this degradation, extra area and power is incurred as well.

These numerous steps in timing closure result in a complicated multi-iteration synthesis flow, the results of which must incur extra area and power. The conceptual diagram in Figure 4.2 shows how these overheads come about. As a circuit goes through the multi-iteration timing checking and optimization, its performance is degraded at each step due to the various sources of variation when compared to its nominal implementation (at the 'nominal' global corner, e.g. TT:V<sub>DD</sub>:27°C). Hold buffers are inserted to ensure hold robustness at each global corner check and derated timing metric check, leading to extra area and power. Since most circuit blocks are

implemented against a specific desired frequency to meet certain application constraints, in an effort to mitigate the speed degradation, designers chose to do logic upsizing to meet setup time, since larger gates have less local variation [6].

### 4.1.2 Hold Failure In Subthreshold

To demonstrate how hold failures occur in  $\text{sub}V_T$ , we performed 3000 point MC simulations on a size X2 hold time friendly standard cell register (Figure 4.3) placed in a shift register (SR) path setting to find the distributions of  $t_{cq}$ ,  $t_{hold}$ , and  $t_{skew}$  (Figure 4.3). Ideal clock and data slew were used. To find what types of skew exist in digital, synthesized blocks, we studied the clock trees of all digital components of a second revision of the ULP BSN SoC in [9] and found a deterministic skew between different register sinks equal to  $0.5\text{FO4}$  delay. Figure 4.3 shows the additional stochastic skew on top of the deterministic amount of skew. Using these results and equations for log-normal distributions, we calculated the hold margin (eq. 4.1) at  $3\sigma$  yield (99.7%) assuming case (b) of Figure 4.3 and found the margin to be negative.

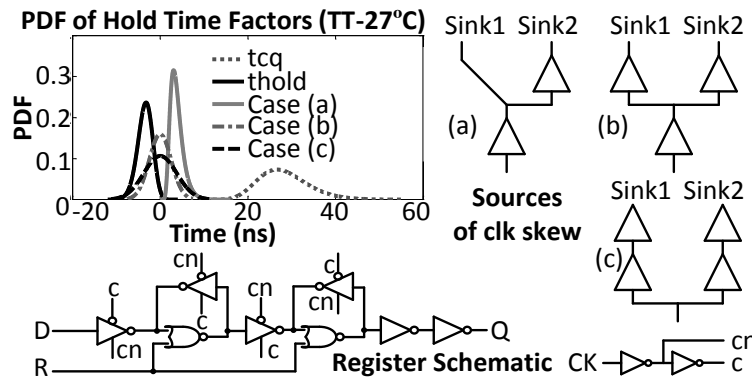


Figure 4.3. MC simulation results of hold friendly register's  $t_{cq}$ ,  $t_{hold}$ , and stochastic skew. Stochastic skew arises from unbalanced clock tree by 1 level (a), or balanced tree but clock paths differ by one level (b) or two (c) [45].

This means that even a hold friendly register presumably protected by its  $t_{cq}$  delay can fail hold time without buffering. It should be noted that two factors, clock jitter and slew, were not

included in this discussion. While it is obvious jitter will make the negative margin worse, slew also contributes negatively through stochastic means [40].

$$\begin{aligned} \text{Hold margin} &= t_{cq} - t_{hold} - t_{skew}(\text{deterministic}) - t_{skew}(\text{stochastic}) \\ &= 15.23 - 2.34 - 12 - 7.9 = -7.01 \text{ ns (TT:27°C corner)} \end{aligned} \quad \text{eq. 4.1}$$

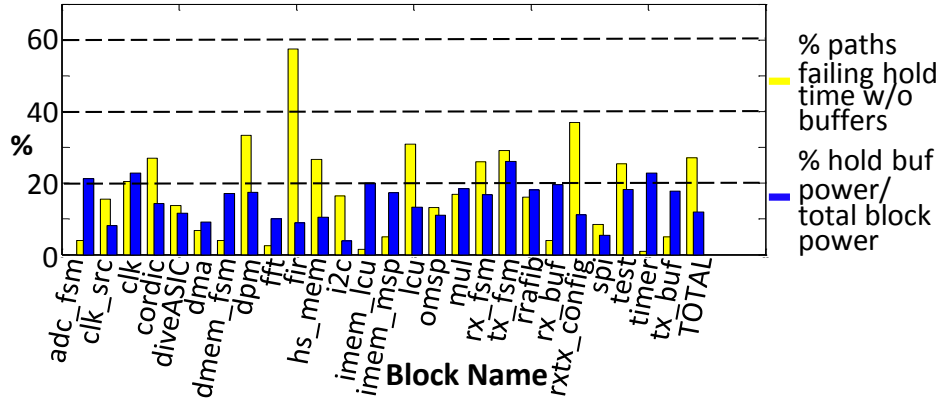


Figure 4.4. Percent paths failing hold time without hold buffers and percent hold buffer contribution to total block power, displaying the significant overhead of conventional hold buffer insertion [45].

Table 4.1. Number of buffers needed to meet hold time per path [45].

	Yield-no buffer	#Buffs Insert	Yield-w/ buffer
<b>Case(b)</b>	89.7%	3	100%
<b>Case(c)</b>	81.3%	4	100%

The severity of hold time in sub $V_T$  is apparent when we took away the hold buffers in the BSN SoC design and used a commercial timing tool to check hold time, with proper timing derates set to reflect the  $3\sigma$  yield for the SoC's technology. The results are shown in Figure 4.4. Figure 4.4 shows that on average a significant portion (12%) of energy is attributed to hold buffers to properly meet hold time. Delving deeper, we found through 3000 point MC simulations of extracted paths (both register and clock path) from the SoCs that for the majority of cases (b) and (c) from Figure 4.3, either 3 or 4 hold buffers must be inserted per path

respectively to meet  $3\sigma$  yield (Table 4.1), which leads to the significant energy overhead. In short, hold buffer insertion *must* be done to meet hold time, which will lead to energy increases.

### 4.1.3 Setup Time Overhead in Subthreshold

To exemplify how OCV affects setup time optimization through logic upsizing, we performed a full synthesis flow implementation of a 16b MAC (multiply-accumulate) unit, and recorded the macro's total area during each step of the flow. First, we performed synthesis at the TT:V<sub>DD</sub>:27°C corner without OCV timing derates or jitter guardband. This step serves as the baseline design, or the 'ideal' design if no sources of variation were present. To isolate the effects of each source of variation, we subsequently closed timing at the worst case global setup corner (SS:0.9V<sub>DD</sub>:0°C) with no timing derates and recorded the resulting area. We then re-ran timing optimization with the design closed at worst case setup corner *with* timing derates and recorded the area. Finally, jitter guardband is added and its area is recorded after successful closure. The results of this experiment, which record the logic upsizing overhead at each step in the conventional timing closure flow, are depicted in Figure 4.5. The resulting area is >2X the original baseline design, thus showing us that logic upsizing is a major overhead.

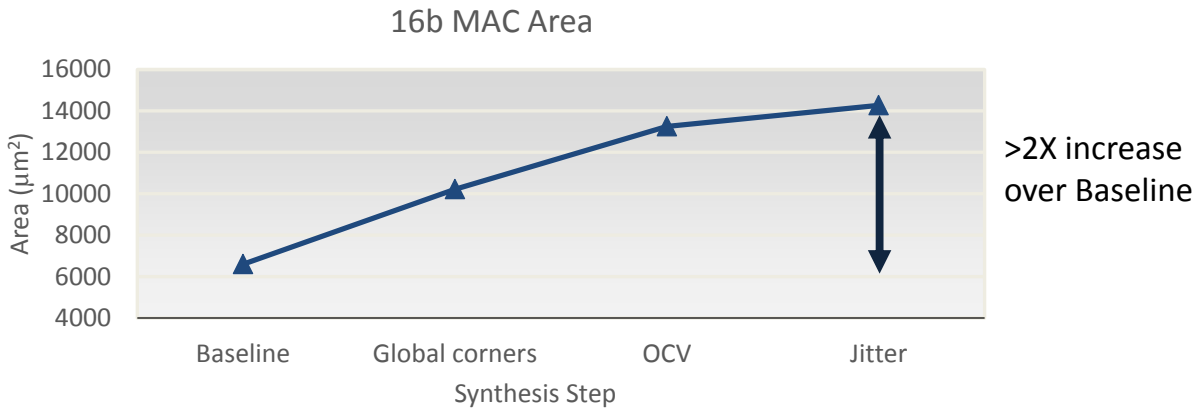


Figure 4.5. 16b MAC area across different steps of timing closure. Global corner and OCV contribute greatly to the logic upsizing that results in >2X increase over baseline design area.

## 4.2 Related Work

Some recent work has focused on fixing the issue of timing closure in  $\text{sub}V_T$ , albeit most of these issues focus on setup closure. Nevertheless, these works represent a growing field of research as designers begin to realize the importance and magnitude of the issue of timing closure. In [47], the authors propose using a soft edge flip-flop where clocks feeding the slave stage of a master-slave flip-flop are slightly delayed, thus providing the flip-flop a short transparency period where data can continue to feedthrough after the latching (rising) edge of the clock. This brief period of transparency leads to improved timing yield in their design, and helps with setup timing closure. They argue that these soft edge flip-flops have advantages over pulse-based flops since a soft edge flip-flop's functional operation will not deteriorate due to local variations as a pulsed-based flop would. The authors also say that their method would be preferred over latch based timing systems since latches incur clock distribution overhead as well as design complexity. In this chapter, we use a latch based timing system, and propose a design methodology that does not increase the clock distribution overhead due to using latches, and explore a synthesis flow that easily integrates latch based design. The soft edge flip flop idea does come with its drawbacks in the form of requiring distributed pulse generators and increased hold time requirement. Of significance is the increased hold time requirement due to the transparency window of the soft edge flip-flops. As our previous sections show, both hold and setup time cause significant overheads in  $\text{sub}V_T$ . Thus, ideas like the soft edge flip-flop that sacrifice hold/setup time robustness in favor of setup/hold time robustness do not offer a complete solution to the timing closure issue in  $\text{sub}V_T$ .

Several solutions have been proposed from a system perspective [48][49][50], again mostly to remedy setup time issues at ultra low voltages. [48] implements an adaptive timing error

detection scheme by inserting special flops on potentially critical paths that minimizes pessimism due to global variations. The flops are able to detect a setup time failure through a speculation window that allows data to feedthrough after the latching (rising) edge of the clock. In this way, as PVT conditions change, the clock period can be adjusted so that the circuit can run at the maximum frequency that the variation dictates. However, there are numerous drawbacks with this approach. Due to the speculation window, again hold time robustness is ignored in favor of setup time. The special flip-flops contain analog components, adding to design complexity both from the circuit and synthesis levels of abstraction. The Razor systems proposed in [49][50] implement a similar approach, though their speculation window and error detection is done with digital circuits. The Razor system approach is throughput based—the authors use supply voltage as a tunable knob to adapt their circuit to a certain throughput constraint. The Razor system has the distinct advantage that setup errors that are detected are dynamically ‘erased’ through architectural replay. However, Razor incurs complicated error detection circuits to implement, and offers no improvement for hold time robustness.

An especially intriguing literature is that of Bubble Razor [51]. First off, they choose to use two-phase latch based timing to implement their variation aware robust system. This system is hold time robustness aware by using two-phase, non-complementary clock phases, though no specific treatment is given as to how to shape the two-phases. Nevertheless, the use of two-phase latch based timing breaks the dependency between short path constraints and speculation window. Thus, setup time robustness optimization can be done without sacrificing hold robustness. In contrast to Razor, which incurs a significant performance penalty when an error is detected by total architecture replay, Bubble Razor allows ‘local correction’ of errors by clock gating all pipe stages by one clock cycle. In this way, an extra clock cycle of time is available to

help the critical path meet its setup time, and no architecture replay is needed. Aside from a decrease in throughput by using this method, the authors do mention that there is an expensive overhead of integrating the Bubble Razor logic to each latch in the design in terms of area and power. To remedy this, complex and specific techniques to perform latch clustering must be done to mitigate the expensive overheads.

As to hold time fixes, LSSD (level sensitive scan design) provides an interesting solution [52]. In LSSD schemes, shift-register paths for scan purposes are robustly designed by using two-phase, non-complementary clocks driving level sensitive latches, such that consecutive latches in a shift register are never clocked (become transparent) simultaneously. In this way, hold errors do not occur. It should be noted that our scheme references [52] and the non-complementary clocks are similar to [52]. At the same time, we emphasize the novelty of our design, which is:

- It is integrated into standard synthesis flows.
- It requires no custom design and is not limited in scope to scan registers.
- The shapes of the non-complementary clocks are post silicon tunable through the availability of a DLL, and do not require user interference during deployment of the circuit.

Our proposed solution will look to improve the overhead incurred in both hold and setup time robustness in the face of both global and local variations. In addition, it will require no additional design/scheme specific circuitry and can be easily integrated into standard synthesis flows with a standard cell library. Finally, our solution offers the advantage of no degradation in performance, and the amount of effort dedicated towards setup or hold time robustness can be post silicon tunable.



## 4.3 Solution

### 4.3.1 Hold Time Closure

Eq. 4.1 reveals that clock skew is a deciding factor in hold time closure. Truly, if we do not consider the skew incurred, the hold margin is positive, even with the effects of OCV apparent. This result is expected, since registers on hold critical paths are designed to provide some inherent hold robustness through double inverters (Figure 4.3) on the output node Q, thus having a long  $t_{cq}$  time. Unfortunately, due to the design of a clock tree, skew is unavoidable in synthesis based designs. With unbalanced routing and unbalanced output load on the clock tree leaf nodes (the nodes directly feeding the clock inputs of registers or latches), deterministic skew is inevitable. Since multiple clock tree buffers are used throughout an entire design to deliver a reasonable clock slew, different clock tree paths to different registers or latches is inevitable, necessary leading to stochastic skew.

Hold time failure due to skew is diagrammed in Figure 4.6. In a register based design, what happens is that skew (both deterministic and stochastic) will move the capturing element's (register or latch) latching edge a delay later than the launching element's latching edge. This is commonly referred to as 'positive skew'. Though positive skew improves setup time closure, it is detrimental to hold time closure. When this arrangement of clock edges occurs, a short logic path consisting of 0-2 logic stages may have a short enough delay so that the data arrives at the capturing element in time to meet the setup time of the capturing element. The probability of this phenomena occurring is increased with the effects of OCV, which may decrease the short logic path's delay further. Thus, hold failures occur often.

Our latch based method uses non-complementary two-phase clocks for alternating pipe stages whose phases inherently deny the possibility of having simultaneous transparency time, eliminating the potential for hold time errors. The proposed method is detailed in Figure 4.6. While latch based timing is a known approach, this is to our knowledge the first application of the method to fixing hold time errors in subthreshold. In this method, each register is split into two latches, so the total number of pipe stages is doubled. The combinational logic block (CLB) between stages is roughly halved and dispersed across the two latch phases. This can be accomplished using standard EDA synthesis tools. Non-complementary clocks feed the two alternating latch phases. The non-complementary clocks are the key to the scheme, as they will be the ones who negate the sources of variation causing hold failure. The whole pipeline's frequency is doubled so that the entire pipeline's throughput remains the same. The two phases we assume for this project are created from a DLL, which are abundant in an SoC environment, and are thus post silicon tunable.

The key to fixing hold time in this timing scheme is the period of dual opaqueness for the latches, which we call the *hold shoulder*. As shown in Figure 4.6, similar in the register base case scenario, the launching element launches the data into the path. Q1 will arrive at the capturing latch. It may arrive quickly with a short  $t_{cq}$  and long  $t_{hold}$  because of variation. The transparent phases of the capturing and launching latch may come close to overlapping due to clock skew. However, regardless of the variation endured, we can always tune the hold shoulder so that the edges do not overlap, and are far apart enough to ensure robust hold closure. Because of the tunable nature of this timing scheme that ensures data will not be latched in the capturing element one cycle early, we believe the two-phase latch based timing scheme will get rid of all need for hold buffering, thus lowering the overhead of robust hold closure in  $\text{sub}V_T$ .

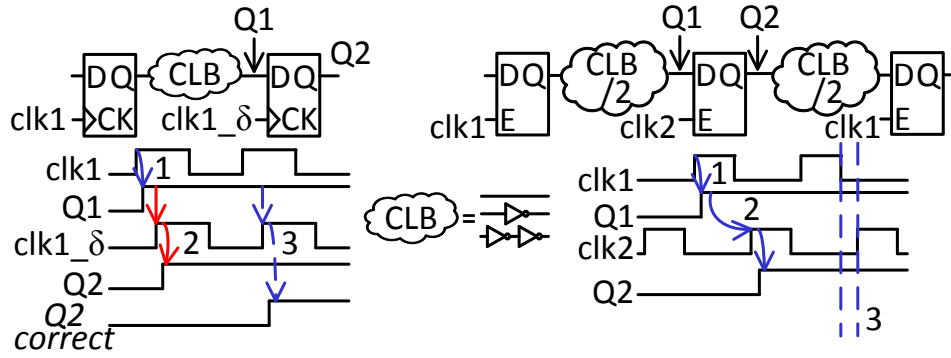


Figure 4.6. On left, diagram of register based hold failure. 1: Data launched to Q1. 2: Skew causes Q1 to meet setup time  $\rightarrow$  hold failure. 3: Desired operation. On right, diagram of two-phase method. 1: Data launched to Q1. 2: Non-complementary phases negate skew even with variation and desired correct operation is ensured. 3: The ‘hold shoulder’ [45].

### 4.3.2 Setup Time Closure

Since we have doubled the number of pipe stages in this scheme, it makes sense to use latches instead of registers in order to lower the overhead of storage elements in the design, as a register is inherently comprised of two latches. In addition to keeping the storage element overhead (in terms of area and power) low, latches also provide an opportunity for setup time closure improvement due to their transparency, which allows time borrowing (Figure 4.7). With time borrowing, critical paths are able to borrow time from later, non-critical pipe stages, thus providing the opportunity for previously unused positive slack on non-critical paths in register based designs to contribute to setup time closure. With time borrowing, the critical paths now have a less stringent timing constraint, and thus, less logic upsizing is needed to lower the stochastic delay in the path due to OCV.

One important note to make about using the two-phase latch based timing scheme for setup optimization is that unused positive slack from the original register base case design must be available for it to be able to save in the overhead for setup time closure. In cases where all logic paths are regularly structured and balanced, for example in FFT designs, using the two-phase latch based method will not improve setup time closure because these designs lack unused

positive slack available for time borrowing when translating to the two-phase latch design. In other words, the critical path must reside somewhat early or near the middle of the entire pipeline, while following pipe stages have shorter, non-critical paths with unused positive slack available. In this way, as the data traverses to the end of the pipeline, time borrowing comes to a closure (no more time borrowing is needed), and the entire pipeline's throughput constraint is met. These concepts are illustrated in Figure 4.8.

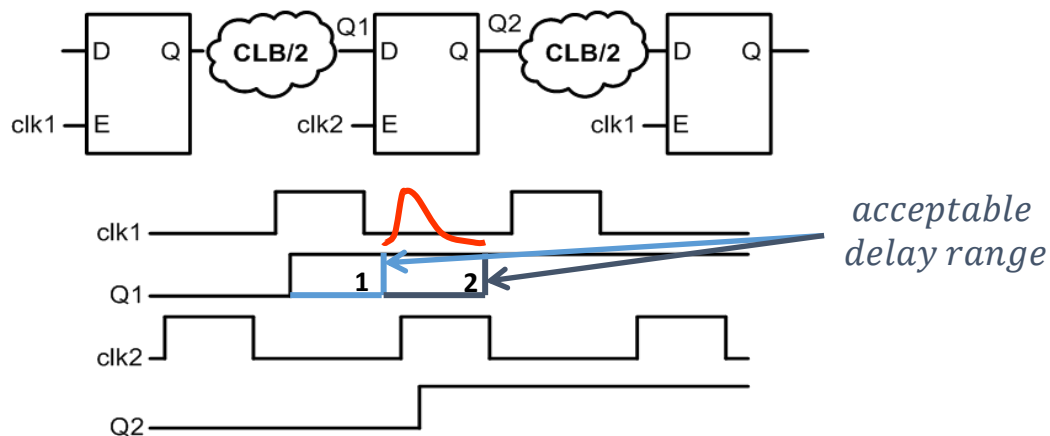


Figure 4.7. In register designs, logic must be excessively upsized so the delay of the data Q1 on a critical path arrives before edge 1. With time borrowing, logic upsizing can be relaxed because Q1 is allowed to arrive anytime between edge 1 and edge 2.

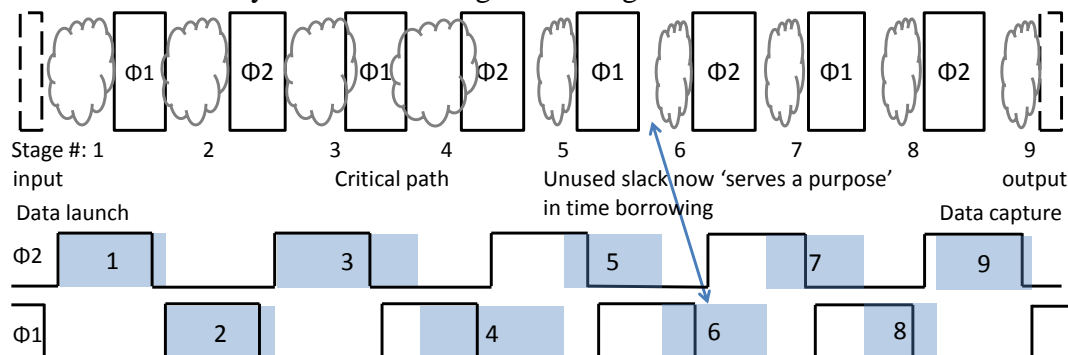


Figure 4.8. Pipe stages 1-2 meet the clock period constraint. Stages 3-4 are critical paths. Without timing borrowing, logic upsizing must be done to lower the stochastic delays of these stages so that they meet the clock period constraint. With time borrowing, stage 3 borrows from stage 4, and stage 4 borrows from stage 5, instead of logic upsizing. Stages 5-9 have less delay than a full clock period, thus allowing time borrowing to recover.

### 4.3.3 Implementation

The implementation of this method has the advantage of being readily integratable with commercial synthesis tools. A register based design is first implemented. The amount of phase offset (the ‘hold shoulder’) between the two phases can be tuned based on results of timing closure and anticipated jitter, and this can occur after fabrication if necessary. A custom script translates all registers into latches, and re-timing is done so that the number of total pipeline stages in the design is doubled, while the logic between pipe stages is roughly split in half. The doubling of pipe stages retains the original throughput of the design, while register to latch translation reduces the storage element energy. The creation of the two phases assumes the availability of a DLL, which are abundant in an SoC environment. In summary, a well defined ‘hold shoulder’ provides resiliency against variation, and the hold shoulder can be tuned based on how much variation is present (how many sigma yield to guard against, how much jitter anticipated, etc.). Figure 4.9 shows a diagram of our implementation flow.

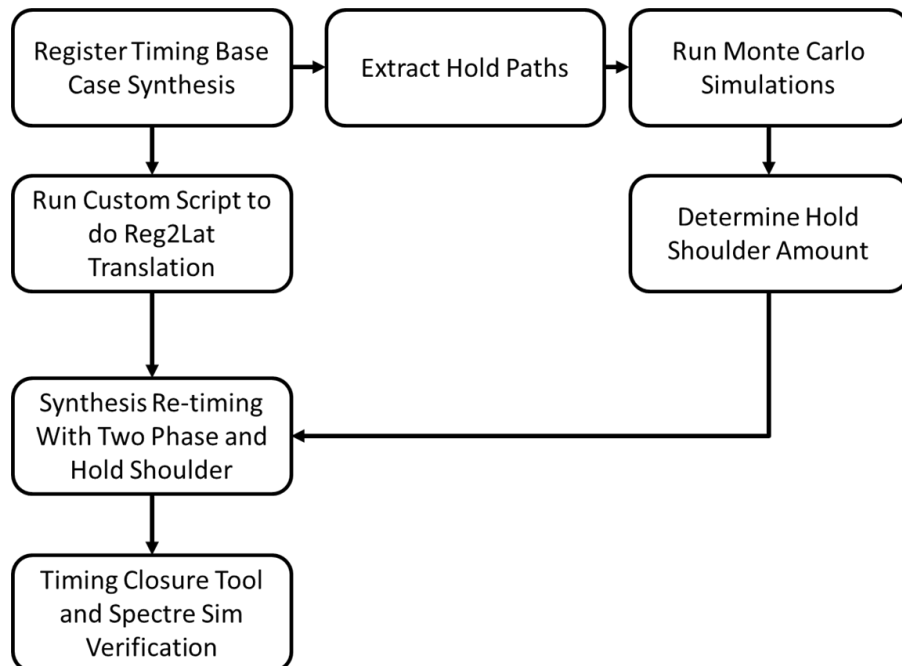


Figure 4.9. Implementation flow diagram for two-phase latch based timing method.

## 4.4 Results and Analysis

The two-phase method was implemented on a 16b, 4-stage MAC, on a 32b, 8-stage shift register (SR), and an 8b, 4-tap, 8-stage FIR. These blocks were chosen to represent the setup time critical/hold time non-critical case (MAC), the hold critical/setup time non-critical case (SR), and a mixed logic path length ‘common case’ (FIR). The straightforward implementation of an FIR provides both hold and setup critical paths, as depicted in Figure 4.10. Several iterations of the designs’ register base case and two-phase implementation were done at the 3-, 4-, 5-, and 6 $\sigma$  yield constraint with a  $V_{DD}$  of 0.3V. Hold time was checked across all global process and temperature corners (SS:0°C, TT:27°C, FF:45°C), and setup time was checked at SS:0°C. We also imposed 1FO4 inverter delay of clock jitter (50ns). These implementations resulted in hold shoulders of 80, 100, 120, 140ns and operation frequencies of 667k, 555k, 460k, 385kHz for 3-, 4-, 5-, and 6 $\sigma$  yield, respectively. None of the two-phase implementations required any hold buffers. A comparison of energy per operation and energy breakdown is given in Figure 4.11.

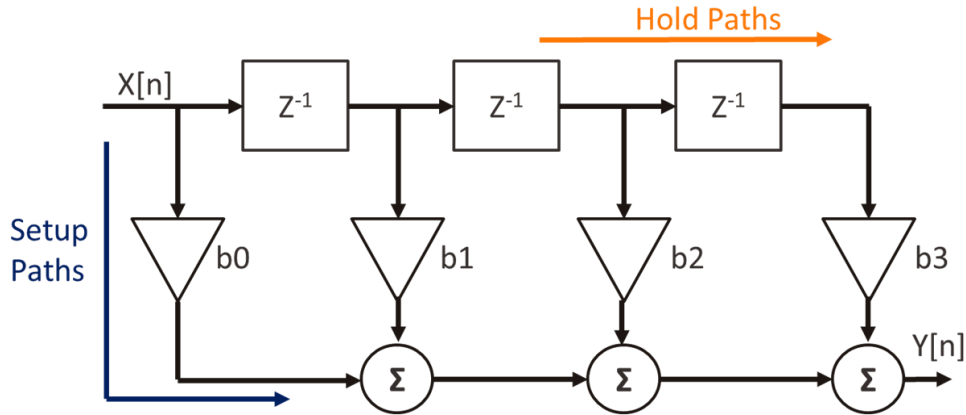


Figure 4.10. Block diagram of implemented FIR, displaying setup and hold paths.

As the yield constraint is made more stringent (from 3 to 6 $\sigma$ ), the savings from two-phase latch based design increase gradually. For the SR, whose savings mainly come from hold buffer insertion negation, additional hold buffer energy increases from 3 to 6 $\sigma$  (Figure 4.11 (b)). The

reason is twofold: the main culprit of skew is case (b) of Figure 4.3, whose distribution is Gaussian, and the early derated delay of hold buffers does not change much from  $3\sigma$  to  $6\sigma$  due to the nature of log-normal distributions. The SR is able to net energy savings of 27-37% compared to the register base case. Special care must be given to the design of the clock trees in the two-phase design. The clock tree should be designed so that the insertion delay (d-q delay, or  $t_{cq}$ ) of a latch is similar to the insertion delay of an original register ( $t_{setp}+t_{cq}$ ). In this way, the sum size of two clock trees for a two-phase design is about equivalent of the one clock tree in a register base case design. This is important because clock tree power can be a significant portion of the total macro's power. Since a register is being split into two equivalent latches, the total clock tree load should remain about the same before and after two-phase translation. In this way, the clock tree power does not significantly increase due to two-phase translation. For the SR, the base case implementation clock tree has a total buffer size of 96, while the two-phase implementation has a total buffer size of 90.

By allowing time borrowing, the MAC is also able to save 45-47% energy. With time borrowing, setup critical circuits are able to use originally unusable positive slack in register designs to meet setup time instead of upsizing and rebuffering. The amount saved is constant across yield constraints, because the clock period, delay of a logic stage, and amount of slack scale evenly from one yield constraint to another (Figure 4.11 (c)). Using two-phase latch time borrowing to improve the overhead of setup time closure is an interesting case. This is because since we have doubled the amount of pipe stages, essentially we have added delay into the logic path due to double the amount of insertion delays. This means that in order for time borrowing to be an attractive option, the amount of unused positive slack must be great enough to overcome the added amount of insertion delay. Using the results of the register base case design, we can

analytically anticipate whether time borrowing will help with setup time closure or not. The formula to do this is in eq. 4.2, which analytically estimates the amount of slack on average per pipe stage after two-phase translation available to be used for time borrowing. Based on the results of this equation and empirical experience, a designer can anticipate if using two-phase latch based timing will have a significant impact on setup time closure overhead.  $t_{slack}$  is the estimated amount of slack per pipe stage after two-phase translation,  $T$  is the clock period after two-phase translation,  $t_{insertion}$  is the anticipated amount of latch insertion, which can be estimated by extracting the insertion delay of registers in the base case design,  $n$  is the number of pipe stages after two-phase translation, and  $t_{logic}$  is the total delay of logic from macro input to output, extracted from the register base case design. In our MAC implementation, both registers and latches had a  $+3\sigma$  yield worst case stochastic delay of  $\sim 300\text{ps}$ . Of 2888 logic paths checked for timing, 17% of those paths time borrowed  $>55\%$  of the clock period, while 19% of paths time borrowed  $<25\%$  of the clock period. Thus, the MAC displays some logic path length diversity, enabling time borrowing to become a tool for less aggressive logic upsizing, and lowers the overhead incurred for setup time robustness.

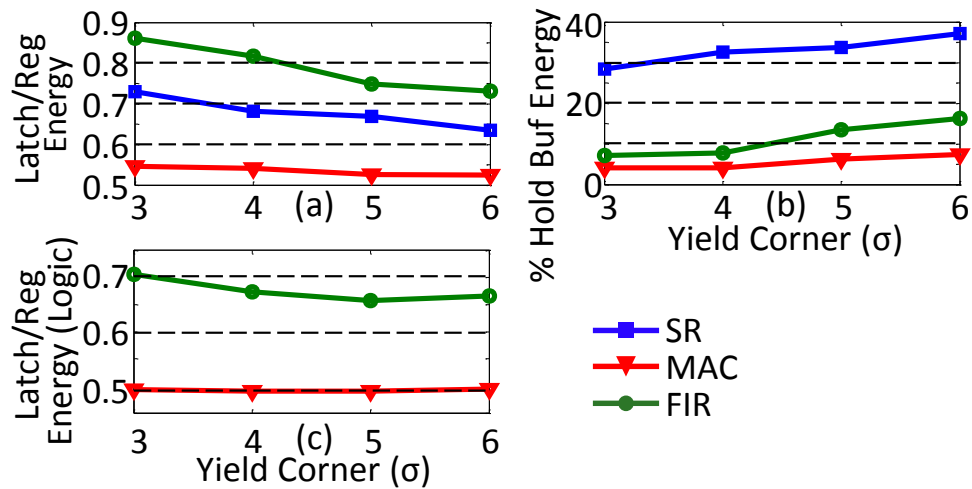


Figure 4.11. Implementation results. (a) Relative energy consumption (latch/register based design). (b) % Hold buffer energy/total block energy for register based designs. (c) Relative energy for logic components (latch/register) [45].



$$t_{slack} = T - t_{logic}/n - t_{insertion} \quad \text{eq. 4.2}$$

The FIR is an interesting case, as it is able to harness both savings from time borrowing for setup optimization, as well as hold buffer negation from the two-phase method. However, these savings are partially compromised due to the latch growth factor (# of latches after two-phase translation/# of registers in base case), which is 2.3 for the FIR, resulting in increased energy in the clock network and latches in the FIR. With all factors accounted, the FIR saves 14-27% in energy per operation across yield constraints. During the design of the FIR we also noticed another key factor to consider for the implementation of a two-phase latch based design, that of the added stochastic skew due to using two different clock trees. In a register based design, a correctly designed clock tree will at most create clock paths differing by 1 or 2 clock tree levels. However, in a two-phase latch implementation with two clock trees, the clock paths differ by as many levels as there are in the clock tree, thus potentially adding to the amount of stochastic skew. Therefore, it makes sense to create the second phase of the clock on a deep level (closer to the latch clock input) of the clock tree to minimize the amount of stochastic skew. Our designs implemented were not large enough to command a clock tree deeper than 2 levels, so the stochastic skews in our designs were not large.

## 4.5 Conclusions

In conclusion, our two-phase, latch based method provides an ULP solution to subthreshold timing closure in the face of PVT variation and mismatch compared to the conventional register based designs that must resort to upsizing, rebuffering, and hold buffer insertion to meet timing. Our extensive simulations show that our method can save up to 37% in energy per operation for hold critical circuits and 47% for setup critical circuits at  $6\sigma$  yield.

Improvements that can be made to the method include expanding the implementation to circuits with feedback logic. The register to two-phase latch translation script so far only supports feed-forward logic, and all implementations done are feed-forward logic only. In order to use this method expansively across all synthesizable designs, the translation script must be updated to incorporate feedback logic as well. Latch overhead must be controlled to further this project, since the two-phase translation leads to increase in the storage element power and area through latch growth factor, and degrades the amount of slack usable for time borrowing because of added latch insertion delay. Care should be given to the design of the two clock trees for the two-phase implementation so as not to drastically increase the clock power or increase the amount of stochastic skew in the design. An alternative to explore is the use of local pulse generators to create the second phase. Work in this chapter has been published in [YQ2] [YQ11].

# **Chapter 5: Logic Family Exploration for Subthreshold Circuits**

## **5.1 Background**

Standard cells are key components to the design process in synthesis, since essentially all digital blocks are comprised of standard cells, and they are involved in every step in the synthesis process. During RTL to gate mapping, the RTL is translated to an assortment of standard cells. During place and route, standard cell layout information enables auto-routing tools to perform its connectivity function. During timing closure, standard cell delay and slew

information provide timing optimization tools their inputs. All of this useful information is extracted from the design of a standard cell. While the design process of standard cells, whether they be basic logic gates (NAND, NOR, INV, etc.) or state saving components (LATCH, REG, etc.) or other ‘gadget’ type components (TRI-GATE, CLOCK-GATE, etc.) is well established for superthreshold, it is largely not optimized for ULV regimes.

### **5.1.1 Overhead of Conventional Static CMOS Libraries**

Simply using industry established standard cell libraries scaled to ULVs will result in imbalanced cell delay metrics for pullup and pulldown operations because the PMOS vs. NMOS strengths change, slowing down the circuit and increasing the penalty of leakage current. Perhaps more vital is the issue of standard cell robustness. PVT variations cause major issues for standard cells. Cells with stacked transistors have a much greater chance of failing SNM and OSV. Thus, it is imperative that conventional standard cell libraries are re-characterized for balance and robustness to PVT variations for  $\text{sub}V_T$ . The passing of SNM and OSV tests is what we mean when we refer to ‘cell robustness’ (Figure 2.5). The issue of scaled libraries into the  $\text{sub}V_T$  regime becoming non-robust is not a totally new topic. [8] raises the issue’s awareness. However, it can also be seen from the literature that the two straightforward remedies to the cell robustness issue come with significant overheads. The first straightforward solution is to scale the industry supplied static CMOS logic family standard cells into  $\text{sub}V_T$  but only to an extent where the cells still retain their robustness to SNM and OSV tests. This has the major drawback that often times the supply voltage at which cells begin to fail is significantly ( $>50\text{mV}$ ) above the minimum energy point (MinE point)  $V_{DD}$ . Therefore, this crude method will often deny designers the opportunity to operate at the MinE point and may drastically increase the circuit’s energy per operation. The second solution, as discussed in [8], is to resize (upscale) cells so that once again

they meet robustness constraints (cells meet a certain yield constraint when SNM and OSV MC simulations are performed on them). The drawback to this solution is that depending on how low the supply voltage is, the cells must be upsized 30-300% above their minimum allowable sizes. Therefore, this method incurs a great area and power overhead. Because of the great power or area overhead of these two straightforward methods, we speculate if industry supplied static CMOS logic family standard cell libraries provide the lowest energy per operation under a certain robustness constraint in sub $V_T$ .

### 5.1.2 Alternative Logic Families

In this chapter, **we address Challenge 5**, and propose a logic family study to find out which logic family provides the most energy efficiency in sub $V_T$  under the same robustness constraints. Different types of logic families have been proposed in the past because they have excelled in one aspect or another over the conventional static CMOS logic family. The term ‘logic family’ is loosely defined, but can be qualitatively asserted as ‘a set of logic gates with similar construction, compatible logic levels, power supply characteristics, and pullup/down mechanisms’.

Clever and exotic logic families making use of special manufacturing materials provide one area of alternative logic families [53][54]. For example, in [54], the authors make use of floating gate devices, thus inherently adding tunable biasing to logic gates. In other words, a transistor’s current draw and drive strength can be modified through controlling an input voltage instead of device sizing, in turn decreasing the transistor area of each logic gate. *Array based* libraries muddle the line between macro and standard cell by implementing commonly performed mathematical operations such as long integer adds, multiplies, and large fan-in AND/ORs as one ‘cell’ [55][56]. These ‘cell’ speeds are drastically increased by pre-charging their output and using a sense-amplifier to determine a logic ‘1’ or ‘0’ output, much like in SRAM arrays

[55][56]. Some logic families minimize energy consumption by implementing *adiabatic logic* [57][58], where the supply rail is connected to a slow changing, multi-phase sinusoidal clock. As logic functions propagate from stage to stage, each gate draws power from the rail when the clock phase compels it to perform its logic function, but subsequently the gate returns power to the supply through capacitive coupling enabled by the sinusoidal clock. In this way, each gate's energy consumption is brought to a minimum through charge recycling, albeit at a penalty of speed.

Since speed is of the utmost importance in super $V_T$ , there are many logic families originally designed for super $V_T$  that look to improve logic delay performance. Three of the most prominent are *bootstrap logic*, *dynamic/domino logic*, and *differential cascade voltage switch logic (DCVS)*. Bootstrapping logic adds a capacitor in series with the input to the gate of transistors [59][60]. As inputs swing high and low, the other end of the capacitor (connected to the transistor gate) receives a boosted high ( $>V_{DD}$ ) or low ( $<V_{SS}$ ), thus speeding up the gate's delay with a higher overdrive voltage, with the penalty of added area and leakage current. Dynamic or domino logic [61][62][63] implements a pre-(dis)charge phase in each gate (thus each gate is clocked). In this way, either the pullup or pulldown time is effectively 0 having been pre-(dis)charged, in turn speeding up the circuit. During the evaluation phase of dynamic/domino logic, each gate conditionally pulls down if pre-charged or pulls up if pre-discharged. Penalties of dynamic logic include added design complexity due to the need of careful handling of a dynamic output node, controlling clock phases, implementing non-monotonic logic (logic that both pulls up and down during evaluation), as well as added dynamic power during the pre-(dis)charge phase. DCVS logic conveniently implements the output of logic as well as its complement in one compact cell [64][65][66]. Since output and output complements are both needed a majority of time in generic,

synthesized RTL, a DCVS gate decreases the logic depth needed through a logic path by simultaneously providing the output complement, thus improving circuit performance. A DCVS gate is built with cross coupled pullup transistors connected to either the gate logic's pulldown network or the gate logic complement's pulldown network (Figure 5.1). Due to the structure of a DCVS gate, other auxiliary advantages it gains is the ability to share transistors between two pulldown networks such as in an XOR2 or XOR3 gate, thus decreasing transistor count and area [64]. Drawbacks of a DCVS gate include current spikes and increased power consumption due to the hysteresis nature of the logic family.

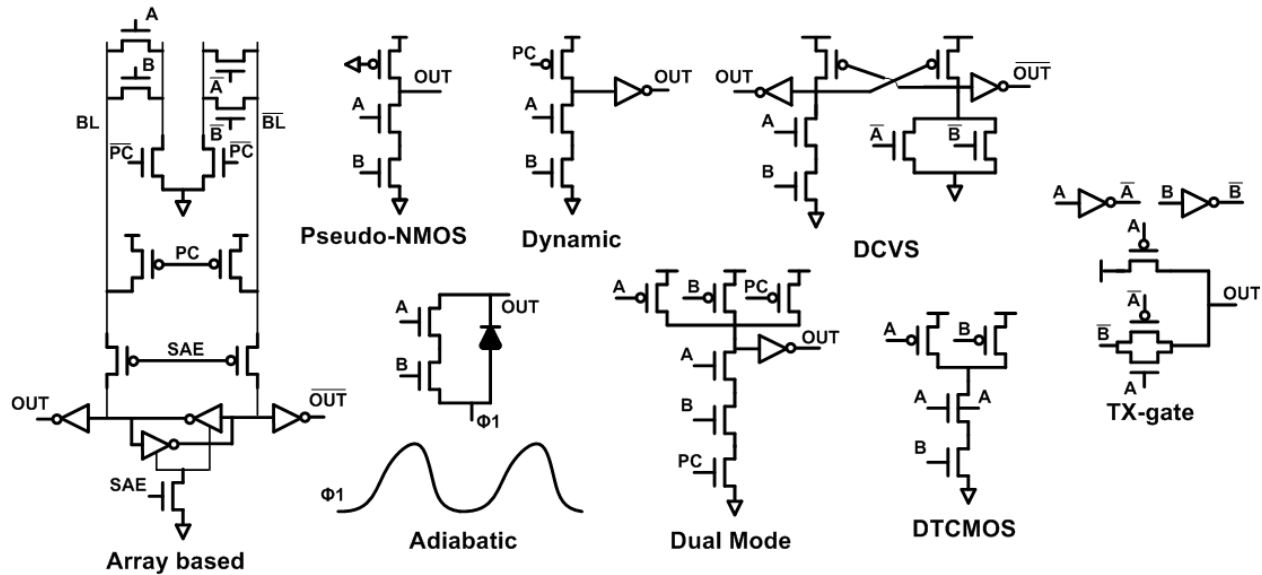


Figure 5.1. Conceptual diagram of various logic families [55][57][61][64][67][69][71][72] considered in this chapter, using AND/NAND gate as an example. *Bootstrap* logic is presented in Figure 5.2.

Lastly, many logic families are designed so they do not differ from the conventional static CMOS style too much, instead creating a new logic family by making modifications to the conventional logic family. An example of this is *pseudo-NMOS* [67], where the complementary PMOS pullup network is replaced by one always on PMOS. The slower and more area consuming pullup stacks are now reduced to one PMOS transistor, but the circuit constantly draws static power. Of special interest are some proposed modifications aimed toward design in

sub $V_T$ . For example, *dynamic threshold CMOS (DTCMOS)* [69] improves cell area, dynamic power, and speed at no cost of leakage power by making the simple modification of tying the body of each transistor to its gate input. By doing so, the threshold voltage of the transistor is modulated as the input changes, decreasing as the transistor turns on. A decreased  $V_T$  during a transistor's on state is analogous to dynamically making the transistor wider, thus decreasing the delay without having to upsize the transistor.

Finally, the *transmission gate (TX gate)* logic family is a unique logic style as it places inputs on the drain/sources of transistors. Nevertheless, this logic family can improve speed, area, and power of certain logic functions such as MUX and XORs. In addition, *TX gate* gates potentially provide added robustness to standard cells. This is because transmission gates use both N- and PMOSs. Thus, the weaker device will always be aided by the stronger device during a pullup/down operation, in turn improving robustness. Table 5.1 gives a summary of these logic families. Figure 5.1 shows conceptual schematics of these different logic families.

Table 5.1. Summary of assessment of different logic families. Red backgrounds mean not suitable for sub $V_T$ . Yellow means not compared in this work but could be viable with a different scope. Green means compared in this work.

Family	Advantage	Drawbacks	Compared?	Refs.
Array	Speed for high fan-in	Extra hardware	No	[55][56]
Adiabatic	Charge recycling	Complicated clocking	No	[57][58]
Bootstrap	Robustness, speed	Large area	No	[59][60].
Dynamic	Speed	Dynamic node, bad robustness	No	[61][62][63]
Pseudo-NMOS	Low area, good speed	Robustness, static current	No	[67]
DCVS	Complementary outputs, speed	Hysteresis	Yes	[64][65][66]
DTCMOS	Speed, power, and area	Larger current spread	Yes	[69]
Dual Mode	Robustness, speed/power tradeoff	Monotonic logic, synthesis difficulty	Yes	[71]
TX-gate	Robustness, speed	Needs rebuffering, complementary inputs	Yes	[72]



### 5.1.3 Context of Subthreshold Synthesis

An important factor to consider when performing our logic family study for robust, energy efficient standard cells in  $\text{sub}V_T$  is its ability to be easily integrated into a synthesis flow using standard EDA tools. The criterion that the logic family is synthesis friendly not only fits within the scope of this thesis (synthesis based design techniques, Figure 1.2), but also lays some ‘ground rules’ for qualitative assessment of the different above mentioned logic families. First, the logic family should be able to be designed in a standard, planar technology process. This is because a change to an innovative process technology would alter the design paradigm too much it would be too hard to make a comparison to conventional static CMOS. Or, the comparison would have to include an assessment of manufacturing flows, which is out of the scope of this thesis. Also, new innovative technologies are not widely distributed, making our comparison infeasible. Further, it is unclear how compatible synthesis EDA tools would be with a new technology. It should be noted that at the time of writing this thesis, promising FinFET technology has not yet been widely distributed as well, and thus, logic families relying on FinFET technology is not compared either [70].

Second, the logic family should require no complex custom clocking schemes or extensive custom design for complicated logic outside the scope of standard cell libraries (see Table 5.3 for a list of common logic implemented by standard cells). Complex clocking schemes, such as though for dynamic or domino logic, often require careful custom tuning during timing closure, which disrupts our scope of a synthesis flow based approach and using standard EDA tools for implementation. Likewise, custom designed ‘cells’ interfere with the cell characterization step and automated timing closure. Custom designed cell layouts that do not lie on a standard cell height grid will incur difficulty during place and route.

Third, given the problem statement at hand (highest energy efficiency under the same robustness constraints), the logic family should offer robustness with respect to both global and local process, voltage, and temperature variations. Since  $\text{sub}V_T$  circuits are inherently slow, leakage current will be an important factor when comparing energy efficiency. The logic family should not induce exponentially increased amount of leakage currents. To make the comparison fair, the logic family should operate strictly in the subthreshold regime. Table 5.2 summarizes the qualitative criteria for our logic family study.

Table 5.2. Qualitative criteria for our logic family study.

Number	Criterion	Synthesis Step(s) Affected	Notes
1	Planar process	All	FinFET excluded in comparison as well.
2	No complex clocking	Timing closure	Prominent in dynamic/domino logic.
3	No complex custom design	Cell characterization, timing closure	Place and route can be affected too.
4	Robustness across PVT variations	All	Both global and local process variations should be considered.
5	No exponential increase in leakage current	Timing closure (power optimization step)	Prominent in bootstrap logic.

## 5.2 Related Work

In this section, we will assess the logic families mentioned in the previous Section 5.1 according to the qualitative criteria established, and explain our decisions as to whether or not to include those logic families in our comparison study. Also, we will assess some recently published work on logic families specifically targeted for  $\text{sub}V_T$ , and will subject them to our qualitative criteria as well.

### 5.2.1 Assessment of Superthreshold Logic Families

All logic families that rely on advantages gained from a non-planar manufacturing process are eliminated from consideration. [53] takes advantage of having a GaAs substrate, meaning

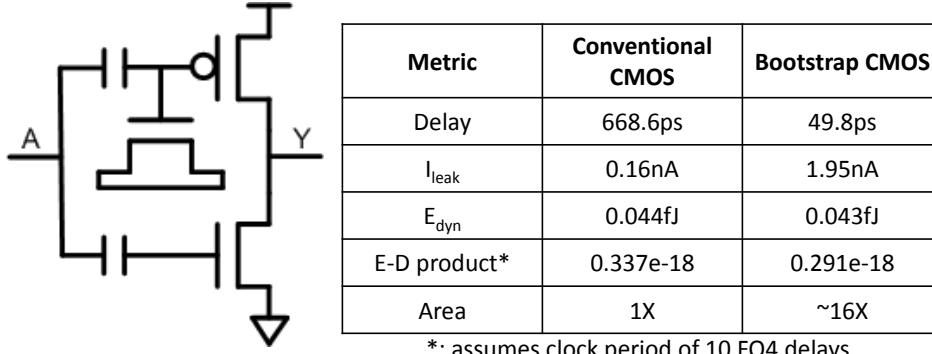
transistors have better carrier mobility than regular planar processes. This makes a fair comparison impossible. [54] makes use of floating gate technology. This literature provides an attractive solution for robustness against global variations. The floating gate devices provide two inputs to the gate of each transistor. The input voltage received by the gate is the sum of the voltage of both inputs. The authors use one input as the logic input, and the other is used as a bias voltage, effectively modulating the threshold voltage of the transistor. This scheme is thus very similar to the body bias circuit technique in [68]. Unfortunately, floating gate technology is not readily widely available.

The *array based* logic families are eliminated from our comparison because they fail criterion 3. Although their PPA (power, performance, and area) metrics are superior for high fan-in functions such as long addition, multiplication, or high fan-in AND/ORing, the overhead to implement lower fan-in, basic logic functions such as NAND2 or NOR2 is great considering the extra hardware needed (pre-charge PMOS and sense amp) to implement an array based gate [55][56]. Instead of designing a full logic family, *array based* designs should be put in the scope of dedicated macros (ASICs, Chapter 2) to utilize their advantages.

The *adiabatic* logic families [57][58] are eliminated from our comparison because they fail criterion 2. In general, for *adiabatic* logic to work, gate supply rails are connected to a slow changing clock, enabling the gate to go through the 3 phases *precharge*, *evaluate*, and *latch* [58]. This causes robustness issues when cascading gates. A gate must not enter *evaluate* phase too early since the preceding gate will have not reached the *latch* state yet (the inputs are invalid), causing the gate to reach an incorrect output value, or have a prolonged delay that does not meet timing constraints. Further, a gate must not enter *evaluate* phase too late since the preceding gate will have already entered the *precharge* phase, thus making inputs invalid. If care is not given to

intricate design of clock phases so that *adiabatic* can operate correctly, robustness of the circuit cannot be ensured. Furthermore, since there are multiple phases, if feedback logic is present, the logic path depth can be no shorter than 3 before feedback can occur [57]. This further limits the *adiabatic* logic family as a viable solution in our context of synthesis.

*Bootstrap logic* [59][60] is eliminated from our comparison because they do not fit the definition of a sub $V_T$  logic family strictly operating in subthreshold. *Bootstrap logic* offers a good robustness solution, since when used in sub $V_T$ , the gate boosting effectively forces the transistor to operate in the linear region. In this way, the transistors are no longer subject to the great sensitivity of OCV effects present in sub $V_T$ . However, this makes their sub $V_T$  operation definition dubious. So, for the fairness of this work, we choose not to compare them. Another drawback is *bootstrap logic's* increased area, due to having to add extra bootstrap capacitors. In other scopes where we do not define a clear criterion for strict sub $V_T$  operation, *bootstrap logic* shows promise to be a leading solution for robust, energy efficient ULV standard cells. Our preliminary experiment shows that the *bootstrap logic* energy-delay product is equivalent to conventional standard CMOS, and it comes with major advantages of operating in the linear region (more robustness to OCV) and drastically decreased gate delays. In our experiment, we compared the energy and delay metrics for a size X1, self-loaded inverter between conventional static CMOS and *bootstrap logic*. Figure 5.2 shows the results. The inverters have the same sizing. As can be seen, the *bootstrap logic* inverter incurs the same dynamic energy, 12X more leakage current for 13.8X improvement in speed. This results in similar E-D products, assuming a clock frequency of 10X FO4 delays of each cell. A major drawback for *bootstrap logic* is its layout area. In order to implement the bootstrap capacitors, MIM caps with large area must be drawn, which causes the cell area to increase by roughly 16X.



\*: assumes clock period of 10 FO4 delays.

Figure 5.2. Schematic of simulated bootstrapped inverter. Simulated comparison results on the right, where the supply voltage  $V_{DD}=0.3V$  [59][60].

*Dynamic* logic families [61][62][63] are eliminated from our comparison due to failing criterion 4. A critical drawback for dynamic circuits is the existence of a dynamic node. Dynamic nodes have little chance of passing robustness tests in  $subV_T$ . As a proof of concept, we ran MC simulations on the output node swing level retention time for both precharged and pre-discharged inverters with X4 size driving an INVX8 gate. Retention time was defined as the time it took for the dynamic node to reach 10%  $V_{DD}$  degradation (30mV). Our results, under these optimistic conditions, show that the  $3\sigma$  yield retention time was 1.5FO4 delays. The worst case retention time occurred at the FF: $V_{DD}$ :45°C global corner. This retention time is far less than acceptable to be viable for consideration to be robust in the face of global variations. A similar MC simulation to test *pseudo-NMOS*'s robustness leads us to discard it too from our comparison.

In the end, our qualitative analysis based on our established criteria leaves *DCVS*, *DTCMOS*, and *TX-gate* logic families available for comparison. Conceptual schematics of discarded logic families can be found in Figure 5.1.

## 5.2.2 Assessment of Subthreshold Logic Families

Some recent literature has proposed logic families especially tailored for  $subV_T$  operation. To the best of the author's knowledge, some of the most prominent include *DTCMOS* [69], *Dual mode logic* [71], and *TX-gate* logic [72].

*DTCMOS* logic gates are conventional static CMOS gates with transistor gates tied to their respective body. As the substrate voltage in *DTCMOS* logic changes with the gate input voltage, the threshold voltage is dynamically changed [69]. In a transistor's off state, its threshold voltage is the same as that of a transistor in a conventional static CMOS gate. As the transistor turns on, its body bias modulates the threshold to a lower (absolute) value, thus dynamically increasing the transistor's drive strength. This leads to two improvement options to the logic gate: 1. high speeds can be attained for the same transistor area with no penalty in active or leakage energy, or, 2. the same speed can be achieved with decreased transistor area and energy. Since energy is a vital metric in sub $V_T$ , option 2 is more attractive. A deeper analysis of *DTCMOS* reveals this circuit style improves gate robustness as well. While the smaller transistor size does not lend improvement to the current spread ( $\sigma$ ) of the transistor, the decreased  $V_T$  value shifts the average of current draw so that the gate's pulldown/pullup is more balanced without having to upsize the gate, thus improving robustness. *DTCMOS* is a good candidate for our comparison, as it provides potential for energy and area savings while ensuring robustness.

*Dual Mode* logic [71] is implemented by adding a pre-charging header or pre-discharging footer connected to the output node of a conventional static CMOS gate. If the header/footer is deactivated, the gate is equivalent to a conventional static CMOS gate. If the header/footer is activated, the logic gate can operate in a fashion similar to *dynamic* logic, thus the name *dual mode* logic. The authors state an achievement of 10X improvement in speed over conventional static CMOS while dissipating 1.5X more power in dynamic mode, or a 5X reduction in power and a 10X penalty of decreased speed over domino logic while in static mode. Dynamic mode in *dual mode* logic is different from conventional *dynamic* logic because there are no dynamic nodes in *dual mode* logic. The pullup/pulldown networks are still fully complementary. Gates are

cascaded in *dual mode* logic by alternating between header- and footer-added cells, so that a logic path resembles that of an *np-CMOS* structure. The fact that logic paths must be custom designed in an *np-CMOS* like structure is a drawback of this idea, and does not fit within the scope of RTL synthesis. The savings results reported by the literature's authors are dubious because only alternating NAND/NOR logic paths were tested. It is not clear if the clock distribution overhead was included in the author's analysis. Nevertheless, there are potential advantages in *dual mode* logic. By retaining a fully complementary pullup/pulldown network, robustness of the cell can be guaranteed, and the circuit will no longer fail due to glitching as conventional *dynamic* logic families do. The addition of a header/footer lowers the sizing requirement of the corresponding pullup/pulldown network. For example, in a NOR2 gate with a header, the stacked PMOSs no longer need to be upsized so their drive strength can flip the output logic level. They now only need to be sized large enough to keep an output logic level '1', as the output will have been precharged to '1' already. Thus, we add *dual mode* logic in our comparison. In order to incorporate it into RTL synthesis, we modify it by discarding the *np-CMOS* structure, instead opting to implement *dual mode* logic with monotonic logic (logic that only pulls up or down during evaluation phase at the output), which makes the logic synthesizable.

The authors of [72] propose *TX-gate* based logic. This logic has advantages in robustness and speed. By using both NMOS and PMOSs in parallel, pulldown and pullup operations are easily balanced, leading to improved robustness, especially that of SNM tests. The parallel of NMOSs and PMOSs also lead to improvements in speed. Take for example, a NOR2 gate. Previously, a pullup transition is subject to two weak PMOSs in a stack. In a *TX-gate* NOR2 gate, the drive strength of the stack is improved with the help of a parallel NMOS, thus making the

beginning of the pullup transition much faster. The authors also point out the advantage that an assortment of common logic functions can be accomplished with one regular circuit template, and only the inputs need to be changed to implement different logic. The work in [72] accomplishes a 16b MAC consuming 0.87pJ energy per operation at 190mV with a speed of 10MHz, thus achieving MHz operation in  $\text{sub}V_T$  while maintaining ultra low power. These results are out of the scope of RTL synthesis though. The authors choose to limit logic depth to no greater than 2, and do not rebuffer the outputs of *TX-gate* gates in between pipe stages. Custom latches also need to be designed for their design to work. The solution proposed is more suitable for custom design. The improvements in robustness and speed entice us to re-examine the *TX-gate* logic family in the scope of synthesis though. To do this, we modify the *TX-gate* solution by creating standard cells designed with transmission gates (in fact, some gates in conventional static CMOS logic family such as XOR2 and MUX2 are already implemented using transmission gates), and create input complements using inverters in each cell to facilitate cell characterization.

A summary of all the assessments made can be found in Table 5.1.

## 5.3 Solution

After selecting the logic families that fit within our scope of  $\text{sub}V_T$  synthesis, we continue to perform the quantitative analysis on them. We reiterate here that our end goal is to determine a logic family suitable for synthesis that accomplishes high energy efficiency while meeting robustness constraints in the form of SNM and OSV tests. The significance of this work is that, to the best of the author's knowledge, no such side by side comparison of several viable solutions has been previously published. Our design process for the logic families consist of the following steps:



1. Select standard cells for design. Design standard cells.
2. Determine and set robustness constraints for the standard cells. Verify which standard cells pass robustness constraints.
3. Characterize standard cells. Compare their energy efficiency both on an individual basis and within the context of a synthesized macro.
4. Compare energy and speed variation of synthesized macro for the different logic families.

### 5.3.1 Standard Cell Selection and Design

Table 5.3 displays which standard cells are designed in our comparison. As can be seen, only a subset of a full standard cell library is designed. This is done in order to save design, verification, and simulation time. We have judiciously chosen the subset so that a majority of cells in a full library are represented by the behavior of similar cells with similar pulldown/pullup characteristics. Cells not covered are candidates for future expansive work.

Table 5.3. Table of cells designed for logic family comparison. Each cell can be representative of the behavior of other cells.

Designed Cell	Represented Cell	Unrepresented Cells
INV	BUF, DLY, CLKBUF	DFF, SDFF, ESDFF, NAND4, NOR4, BUFZ, LAT, CLKGATE
NAND2	AOI211, AOI221, AOI222, OAI21, OAI22, OR6	
NAND3	ADDF, AOI31, AOI32, OAI211, AND6	
AOI21	ADDF, AOI22	
MUX2	N/A	
NOR2	AOI31, AOI32, OAI21, OAI22, OAI211, AND6	
NOR3	ADDF, AOI211, AOI222, OR6	
XOR2	N/A	
XOR3	N/A	

The MUX2, XOR2, and XOR3 gates do not represent any cell structure other than their own. This is because they are the only cells in a full library that are inherently designed in a *TX-gate* logic style. In the table, only the inverting versions of the cell names are listed. For example,

AOI211 is listed, but AO211 is essentially the same cell as AOI211, the difference only being an extra inverter. The full adder cell (ADDF) is commonly implemented as a mirror adder in full libraries, thus having similar characteristics to NAND3, NOR3 (on the sum output path), and AOI21 gates. AND6 and OR6 are commonly implemented as two stage NAND3/NOR2 or NOR3/NAND2 gates, as in eq. 5.1.

$$\begin{aligned}
 Y(\text{OR6}) &= A + B + C + D + E + F = (A + B + C) + (D + E + F) \\
 &= \overline{\overline{(A + B + C)} + \overline{(D + E + F)}} = \overline{\overline{A + B + C} \cdot \overline{D + E + F}} \\
 &= \text{"2 NOR3 gates + 1 NAND2 gate"}
 \end{aligned}
 \tag{eq. 5.1}$$

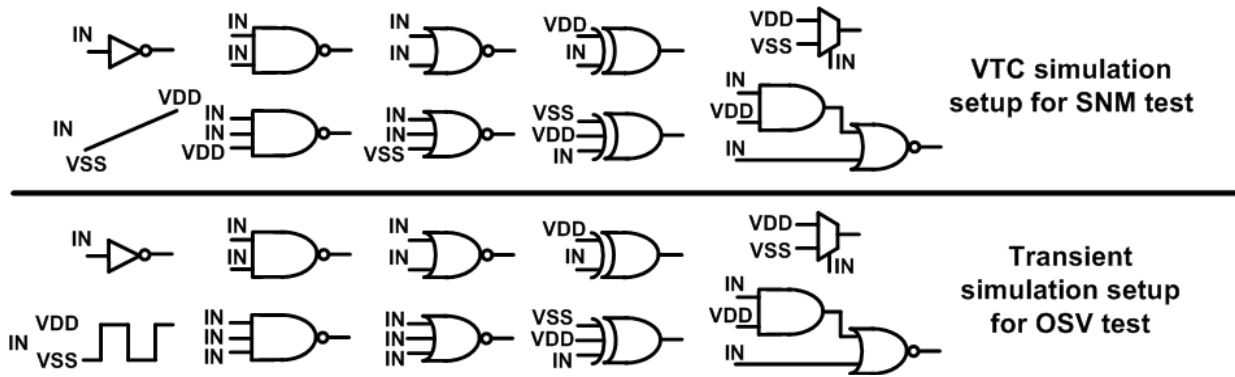


Figure 5.3. Simulation setups for SNM and OSV tests. VTC curves extracted for SNM test. Output logic levels sampled for OSV test. Each gate is loaded by 4 replicas of itself.

We start our design with the conventional static CMOS logic family, which is our reference family for comparison. For our 65nm technology, at a supply voltage of 0.3V with process balance of 2.1 (P/N transistor size for equivalent pullup/pulldown time in an inverter), each cell was designed so their  $V_M$  value was within 5mV of 0.15V at the TT:27°C corner, thus maximizing robustness to SNM tests across global corner variation. Drive strengths of X1, X2, X4, and X8 were designed, with the exception of NOR3X8, which required too great an area and dynamic power overhead in the pullup devices to merit consideration. When running VTC simulations for SNM tests, NOR2, AOI21, and NAND2  $V_M$  was balanced for the worst case input pattern (both inputs are swept). NAND3 and NOR3 gates were balanced when only two

inputs were simultaneously swept, which is not the worst case. This is because we found that if these 3-stack gates were balanced for the worst case input pattern, the devices would become too large to merit consideration. It was found that the worst case VTC for MUX2 is from sweeping the select input (S). The full VTC test setup for SNM tests is presented in Figure 5.3. Subthreshold logical effort conclusions from [44] aided in arriving at the sizing to balance  $V_M$ . Cell speed and area were then improved on a secondary basis after ensuring  $V_M$  was within 5mV of 0.15V.

This design process was reiterated for each logic family. When designing *DCVS* gates, it is extremely difficult to balance the  $V_M$  with different input patterns due to the hysteresis in the gate. Instead, we modify the criterion of design for *DCVS* to balancing the average  $V_M$  between pullup and pulldown transitions (accomplished in a VTC simulation by sweeping the input both from  $V_{SS} \rightarrow V_{DD}$  and  $V_{DD} \rightarrow V_{SS}$ ). When designing *DTCMOS* gates, only the transistors that are in a stack and source/drain not connected to rail have their bodies connected to gate. This is done to reduce leakage in the stack [73], which may increase if the rail connected transistor is on but the stack is otherwise off (think of a NAND2 gate where the bottom NMOS is on, but the top NMOS is off, like drawn in Figure 5.1). The design of *dual mode* gates reveals other drawbacks of this idea. In order to fully exploit their speed enhancements in a synthesis environment, *dual mode* gates must be designed to prevent glitches from happening, or the speed up attained from precharging the internal node is negated. This means *dual mode* gates must be monotonic logic. Thus, this eliminates MUX2, XOR2, and XOR3 gates from being designed in this logic family, since they cannot be guaranteed not to glitch. This is a major drawback, because these three gates are among the most logically efficient gates in a full standard cell library (in essence, they implement an invert, 2 or more AND2, and an OR2 function). Furthermore, because we chose to

speed up the slower PMOS pullup stacks with a precharge PMOS (as drawn in Figure 5.1), the design of a *dual mode* inverter incurs an isolation PMOS, creating a 2-stack in the pullup network of an inverter in order to maintain logic monotony. The induced PMOS stack in an inverter makes the basic inverter gate an expensive entity in terms of area and dynamic power.

### 5.3.2 Robustness Tests

After completing the design of all cells and drive strengths, we subject the cells to robustness tests. The global corners we used for running 500 point MC simulations are TT:27°C, SS:0°C, FF:45°C, SF:0°C, and FS:0°C. All  $V_{DDs}$  were set to 0.3V, which is roughly 30mV above the MinE point  $V_{DD}$  for our 65nm technology. For all cells, their two worst case VTCs are extracted from the results of MC simulation (the most right-shifted and most left-shifted VTC curves). Starting from all X1 size cells, in order for a cell to pass SNM robustness, its worst case VTC curves must form two enclosed areas with all other cells' worst case VTC curves [8]. If a cell fails to do this, its current drive strength is discarded, and the cell is upsized until it meets the SNM robustness requirement. If a size X8 cell still fails SNM robustness, the entire cell is deemed a failure and discarded entirely. After it has been determined which sizes meet the SNM robustness requirement, acceptable output swing voltage levels are determined from the VTCs by finding  $V_{IL}$  and  $V_{IH}$  in the conventional sense [74]. Afterwards, the  $V_{IL}$  and  $V_{IH}$  values are compared against the results of OSV MC simulation results. Cells that have greater OSV degradation than acceptable by  $V_{IL}$  and  $V_{IH}$  standards are discarded. If a size X8 gate still fails OSV robustness, the entire cell is discarded. It was determined that  $V_{IL}=16mV$  and  $V_{IH}=291mV$  to ensure robustness across global and local variations. Figure 5.4 summarizes our entire robustness constraint verification flow using NAND2/NOR2 gates as an example, while Table

5.4 shows the resulting accepted cells that will form our reduced standard cell library and move on to cell characterization to determine energy efficiency.

Results indicate that the most troublesome gates are XOR2, XOR3, and INV gates. INV and XOR2 gates tend to fail OSV at fast NMOS corners that degrade the output high level. XOR3 gates tend to fail SNM at SS and FF corners due to not having enough gain through the 3-stack transmission gate structures in a XOR3 gate which lead to logic level ambiguity in the face of local variation. This is also the reason that the *DTCMOS* MUX2 gate must use a larger drive strength (X4), since a MUX2 gate is designed with a similar pass gate structure, and device sizes are smaller in *DTCMOS*, leading to more local variation.

Table 5.4. Summary of results of robustness tests. Numbers show minimum size of each gate that passes robustness tests.

Logic Family/Gate	INV	NAND2	NAND3	AOI21	MUX2	NOR2	NOR3	XOR2	XOR3
CMOS	2	2	1	1	1	1	1	2	8
DTCMOS	4	1	1	1	4	1	2	2	8
Dual mode	2	2	2	2	N/A	2	2	N/A	N/A
TX-gate	2	1	1	1	1	2	2	2	4
DCVS	N/A. See Section 5.4.								

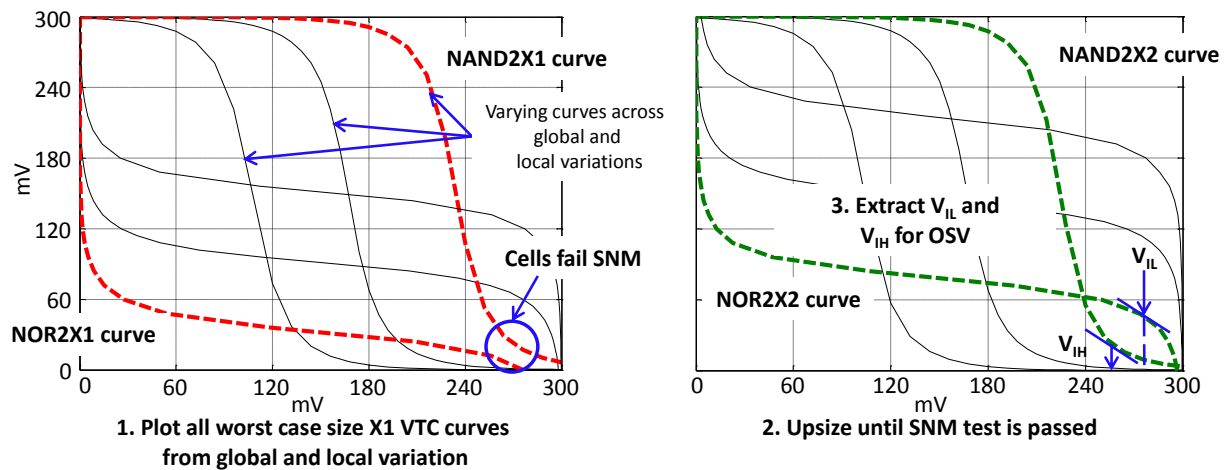


Figure 5.4. Example procedure for standard cell SNM and OSV robustness tests using NAND2 and NOR2 gates. In practice, all gates' VTCs and OSVs are checked.

Our procedure for determining robustness not only ensures that the cells chosen for characterization and use for synthesis will operate robustly, it subjects all logic families to the exact same robustness constraints. Thus, it provides for a fair side-by-side comparison of different logic families. In the next section we will discuss the results of our energy efficiency comparison for the different logic families, and in turn arrive at a conclusion about which logic family to use for sub $V_T$  synthesis.

## 5.4 Results and Analysis

We compare energy efficiency in three areas: E-D product of individual standard cells, energy efficiency of synthesis results, and energy and speed variation of synthesis results.

### 5.4.1 Cell Characterization and Cell Energy Efficiency

. Table 5.5 shows the ‘cost’ metric of each individual standard cell. We define the ‘cost’ metric for each cell as the E-D product when the cell is loaded by 4 INVX2\_CMOS gates and where delay is the slowest arc of the cell amongst all input patterns. Numbers in Table 5.5 are normalized to the INVX4\_CMOS gate cost footprint.

*DCVS* gate numbers are not reported in the table because this logic family was eliminated from comparison at this stage of comparison. Due to hysteresis, *DCVS* gate delays are >3X that of *CMOS*, and energy is >10X that of *CMOS*. A quick synthesis of an FIR macro, which we will eventually use to compare energy efficiency, using *DCVS* gates shows their advantage of implicitly having complementary inputs only decreases the critical path logic depth by 80%. Thus, the evidence is preponderous that *DCVS* gates should be discarded, and they will no longer be part of the discussion. Some cells have reported ‘0.00’ numbers for cost. These are cells that were not designed. For *dual mode* logic, cells MUX2, XOR2, and XOR3 which do not retain

logic monotony aren't designed or characterized. There is no such thing as a *TX-gate* inverter. *TX-gates* with 3 or more inputs cannot be recognized by EDA cell characterization tools, with the exception of the XOR3 gate. Cells that weren't designed or cannot be characterized are replaced by their *CMOS* gate counterparts in a full .db/.lib timing library for synthesis.

Table 5.5. Summary of normalized cost metric (individual cell E-D product) of cells. Some cells that cannot be incorporated in synthesis are given 0.00 values. *Dual mode* gate numbers include delay and energy of added inverter to retain logic monotony. *TX-gate* gates include overhead of inverters to create complementary inputs.

Logic Family/Gate	INVX4	INVX8	NAND2 X2	NAND2 X4	NAND2 X8	NAND3 X2	NAND3 X4	NAND3 X8
CMOS	1.00	1.06	1.66	2.30	3.25	2.24	2.85	3.76
DTCMOS	0.73	0.79	1.53	1.99	1.99	1.85	2.21	2.34
Dual mode	1.32	1.32	1.76	2.68	6.50	3.54	4.00	7.55
TX-gate	0.00	0.00	3.40	5.21	6.72	0.00	0.00	0.00
Logic Family/Gate	AOI21 X2	AOI21 X4	AOI21 X8	MUX2 X4	MUX2 X8	NOR2 X2	NOR2 X4	NOR2 X8
CMOS	6.00	5.35	6.88	4.61	6.97	2.66	2.95	4.24
DTCMOS	4.89	4.18	4.61	3.40	4.39	2.31	2.49	3.30
Dual mode	8.62	10.28	16.46	0.00	0.00	8.19	10.26	18.40
TX-gate	0.00	0.00	0.00	4.61	6.97	3.40	5.21	6.72
Logic Family/Gate	NOR3 X2	NOR3 X4	XOR2 X2	XOR2 X4	XOR2 X8	XOR3 X8	Notes	
CMOS	3.72	4.29	3.15	3.40	5.78	13.39	Reference	
DTCMOS	2.79	3.11	2.42	2.63	4.15	9.61		
Dual mode	11.15	13.71	0.00	0.00	0.00	0.00	Gate+INV	
TX-gate	0.00	0.00	3.15	3.40	5.78	19.93	Extra INV overhead	

*DTCMOS* displays good promise as it has lower cost for all cells compared to conventional static *CMOS* gates. It remains unknown if *dual mode* logic will perform well in synthesis because this logic family's cost numbers include the overhead of an added inverter at the output to ensure monotonic logic. Thus, their delay is essentially premised on two gate delays. *TX-gate* gates have good performance metrics over their *CMOS* counterparts, but suffer in their overall cost metric due to the need of adding inverters for each input to create the input's complement. We are compelled to do this in order for *TX-gate* gates to be characterizable by the EDA tools. If

we create complements on the output of the gate, this would expose a resistive load from the drain/source of a transistor in a transmission gate to the preceding gate, thus making the gate uncharacterizable. Thus, the remaining candidates for comparison are *CMOS*, *DTCMOS*, *dual mode*, and *TX-gate* logic families.

### 5.4.2 Synthesis Nominal Energy Efficiency

We synthesized a 16b, 4 tap, 4 pipeline stage FIR in the different logic families, and compared their operating frequency, leakage current, and total energy per operation. Area was not compared because full cell layouts were not implemented to save design time. All synthesis designs used the same DFF from the foundry supplied standard cell library. All logic families were characterized at the TT:0.3V:27°C corner and synthesis designs used the resulting library. To find the most energy efficient synthesized design, the frequency constraint was swept at 50ns intervals for each logic family and the highest reported energy efficient design was accepted for further spice simulation. Though the method of only synthesizing one macro to compare logic families lacks sample size, the FIR provides a diverse set of logic path depths from hold critical to setup critical as mentioned in Chapter 4, thus lending credibility to results. Table 5.6 reports the pertinent metrics for comparison from this experiment, which represents the nominal energy efficiency of the different logic families.

Table 5.6. Results of energy efficiency comparison of logic families from synthesis results.

Logic Family	Leakage current	Period	Dynamic energy	Total energy/op	E-D product
CMOS	10.7 $\mu$ A	500ns	1.61pJ	3.21pJ	1.61e-18
Dual mode	13.7 $\mu$ A	500ns	9.2pJ	11.3pJ	5.63e-18
PONLY	11.2 $\mu$ A	500ns	1.68pJ	3.36pJ	1.68e-18
TX-gate	10.3 $\mu$ A	500ns	1.43pJ	2.97pJ	1.49e-18
DTCMOS	7.46 $\mu$ A	500ns	0.86pJ	1.98pJ	0.98e-18
NONLY	4.36 $\mu$ A	500ns	0.66pJ	1.32pJ	0.66e-18



It is apparent that *dual mode* logic is not a good idea for  $\text{sub}V_T$  synthesis. Its speed advantage cannot be utilized because it is nearly impossible to automate mapping of RTL to monotonic logic. Roughly 40% of all cells used in the synthesized *dual mode* FIR are INV cells. This means that the gates will still glitch, and the pullup times of the gates still contribute to delay instead of being 0. What's more, this drastically increases the dynamic energy, leading to the disastrous energy efficiency of *dual mode* logic in the scope of synthesis. It should be mentioned that the clock distribution power is minimal to the total contribution of power. Due to the inverter overhead in *TX-gate* cells, they only save 8% in energy efficiency compared to conventional *CMOS*, and because of the extra delay of the inverters, can only accomplish equivalent performance as *CMOS*. As expected, *DTCMOS* is superior in leakage and dynamic energy while accomplishing the same frequency as *CMOS*. It accomplishes 40% savings in energy efficiency.

One of the observations we made during the synthesis of *CMOS* was that NOR2 and NOR3 gates were excessively more expensive in cost metric (>30%) and area (>40%). Using this observation and knowing that synthesis tools are capable of logic restructuring, we added two more 'logic families' *NONLY* and *PONLY*. The *NONLY* family is the *CMOS* logic family stripped of all NOR\* gates, while the *PONLY* family is *CMOS* stripped of all NAND\* gates. We hypothesize that *NONLY* can also incur energy efficiency savings and *PONLY* will incur energy efficiency penalties, and are correct.

The result that a very energy efficient standard cell library can be achieved simply by pruning out 'expensive' cells (*NONLY* even outperforms *DTCMOS*) in  $\text{sub}V_T$  can be dubious, as we cannot be sure if the savings are the result of the synthesis tool not being 'smart' enough to optimize out expensive cells on its own, or a result of logic family design. To investigate the

cause of our 59% energy savings from *ONLY* logic family, we replicated the same experiment for super $V_T$  synthesis. We synthesized the FIR at  $V_{DD}=1.0V$  with the same subset of standard cells that were foundry designed using *CMOS*, *PONLY*, and *ONLY* logic. Both the *PONLY* and *ONLY* designs incurred ~30% decrease in energy efficiency and ~40% increase in area compared to the *CMOS* design. Studying the cells, we found that P/N strength ratio (1.3) is much closer to 1 at a supply voltage of 1.0V, leading to NOR and NAND gates to have similar leakage and dynamic power footprints. NOR gates are still slower by ~20% in super $V_T$ . With these observations we speculate that it is a combined effort of both the EDA tools and logic family design that achieve 59% energy savings for *ONLY* logic in sub $V_T$ . We speculate that the EDA synthesis tool optimizes timing, power, and area optimization by doing cell drive strength swapping while the logic structure remains frozen. For *CMOS* logic family, the synthesis tool asserts that the most efficient logic structure contains NOR/OR constructs, compelling the tool to map NOR gates. This results in an inefficient design in sub $V_T$  where P/N is unbalanced (4:1 sizes in our NOR2 gates), but an efficient one in super $V_T$  where P/N is balanced (1.3:1 in foundry NOR2 gates). For *ONLY* logic family, the synthesis tool can no longer map to what it thinks is the most efficient logic structure, thus it must perform logic restructuring to a less efficient logic structure. This compels the tool to use more efficient gates in sub $V_T$  that improve energy efficiency despite the less efficient logic structure. Indeed, the logic structure inefficiency is exemplified in both sub $V_T$  and super $V_T$  synthesis results, where the gate count increased 12% when switching from *CMOS* to *ONLY* logic.

### 5.4.3 Energy and Speed Variation

To measure each logic family's robustness to variation on a synthesis level, we extracted the MAC unit within each FIR and ran MC simulations on them and measured their delay and

energy. A MAC unit was chosen instead of the full FIR to save simulation time. From Table 5.6 we will only consider *CMOS*, *TX-gate*, *DTCMOS*, and *ONLY* logic families. Table 5.7 shows the  $\sigma/\mu$  values for the *CMOS* logic family. It can be seen that *CMOS* logic family energy varies less than performance, with the exception of the FF:0.3V:45°C corner. Table 5.8 shows the relative  $\sigma/\mu$  values for the different logic families. The *TX-gate* logic family is eliminated from consideration at this step, as it has a significantly larger  $\sigma/\mu$  of speed and energy per operation at the FF corner. *DTCMOS* logic has increased variation, especially at the SS corner, due to having smaller sized transistors. *ONLY* has similar variation to *CMOS* since it is essentially a subset of gates from the *CMOS* logic family.

Table 5.7.  $\sigma/\mu$  values of *CMOS* logic family across different global corners.

$\sigma/\mu$ of Speed(ns)	TT:0.3V:27°C	SS:0.3V:0°C	FF:0.3V:45°C	FS:0.3V:0°C	SF:0.3V:0°C
CMOS	0.101	0.218	0.008	0.078	0.202
$\sigma/\mu$ of Energy(pJ)	TT:0.3V:27°C	SS:0.3V:0°C	FF:0.3V:45°C	FS:0.3V:0°C	SF:0.3V:0°C
CMOS	0.022	0.014	0.006	0.012	0.014

Table 5.8. Relative  $\sigma/\mu$  of different logic families. All values are normalized to the *CMOS* case.

$\sigma/\mu$ of Speed	TT:0.3V:27°C	SS:0.3V:0°C	FF:0.3V:45°C	FS:0.3V:0°C	SF:0.3V:0°C
CMOS	1	1	1	1	1
TX-gate	1.115	1.264	8.330	2.040	0.960
DTCMOS	1.063	1.348	1.854	1.073	0.981
ONLY	0.877	0.930	0.981	0.937	0.901
$\sigma/\mu$ of Energy	TT:0.3V:27°C	SS:0.3V:0°C	FF:0.3V:45°C	FS:0.3V:0°C	SF:0.3V:0°C
CMOS	1	1	1	1	1
TX-gate	1.083	1.006	2.054	1.104	0.804
DTCMOS	1.132	1.348	0.913	1.183	1.105
ONLY	0.983	1.022	0.737	1.216	0.961

This analysis leads us to conclude that *DTCMOS* and *ONLY* logic families are the most suitable logic families to perform sub $V_T$  synthesis with as they provide the best energy efficiency while passing robustness constraints.

## 5.5 Conclusions

Using our method of performing qualitative analysis, robustness verification, and energy efficiency comparison, we arrive at the conclusion that *DTCMOS* and *ONLY* logic families are most suitable to perform sub $V_T$  synthesis with. Under the same robustness constraints, *DTCMOS* displays energy efficiency savings of 40% over conventional static *CMOS* when synthesizing an FIR macro at a sub $V_T$  supply voltage of 0.3V. Due to smaller sized transistors in *DTCMOS*, it has increased variability in delay and energy. Also subject to robustness constraints, *ONLY* logic displays energy efficiency savings of 59% when synthesizing the FIR with similar variability metrics compared to *CMOS*.

Though both logic families are viable alternative solutions to *CMOS* logic with energy efficiency advantages, some conditions exist that may sway us to favor one or the other. When using the *DTCMOS* logic family, the increased variability in speed and energy must be taken into account. Also, if *DTCMOS* is to be used to boost the drive strength of NMOSs (meaning NMOSs were weak to start with), triple-well technology must be available in order for individual NMOS gates to be tied to their individual bodies. Finally, our analysis does not consider layout area. However, if layout area is considered, *DTCMOS* will incur great area overheads, which are caused by individual N(P)well spacing DRC rules to incorporate gate-body tying. For example, a NOR2X1 gate in *DTCMOS* has 2.55X the area of a *CMOS* NOR2X1 gate in our 65nm technology when drawing a layout passing all DRC rules and maintaining standard cell height. For *ONLY* logic, their advantages are gained due to specific behavior of synthesis tools. In cases where synthesis tools are not bound to a certain logic structure during the last stages of timing, power, and area optimization, there would be no need for *ONLY* logic as the tool will freely optimize the netlist without designer interference.

Furthermore, if the scope of  $\text{sub}V_T$  synthesis were changed, other logic families such as *array based* and *bootstrap* logic families could become viable solutions as well. As previously mentioned, *array based* cells can be utilized as auxiliary cells within a full standard cell library, where the *array based* designs perform specific, high-fanin functions. Robustness of *array based* cells can be characterized in ways similar to SRAM columns, as their operation is similar. If application requirements for speed become a compelling factor, a good strategy for energy/speed tradeoff would be to use *bootstrap* logic, where dynamic energy is still equivalent to levels of  $\text{sub}V_T$  operation and speed is improved due to transistors operating in the linear region.

Finally, we take note that our comparison presented in this chapter does not provide 100% coverage of the behavior of all standard cells in a full library, most glaringly registers and latches. Research for robust, energy efficient design of these cells is a top candidate for future work.

## Chapter 6: Conclusions

Recent emerging application spaces including WSNs and BSNs call for ultra low voltage designs, as these circuits can harness the dynamic energy savings from designing in the  $\text{sub}V_T$  regime to meet application ultra low power requirements. Impeding more wide spread acceptance of ULV circuits is the issue of their robustness, which includes such topics that range from device lifetime, to packaging hardness, to radiation hardness, to timing closure, to correct circuit operation. This dissertation successfully addresses a subset of those issues, which are device lifetime, timing closure, and robust standard cell operation in hope that ULV circuit may gain wider acceptance and usage in the near future.

Outside the scope of ULV design, the broader impact of this work is that as high performance circuits continue to be limited by the power wall, the scaling down of  $V_{DD}$ , and increased OCV as device sizes continue to shrink, synthesis based robust and high energy efficiency circuit design techniques will increasingly become a key to continued innovation in the VLSI world. Intel's Claremont processor [75] runs at a near-threshold voltage of 0.5V, thus already propelling innovation of energy efficient digital VLSI to the ULV regime.

## 6.1 Summary of Contributions

This dissertation contributes to the cause of robust ULV circuit design with the following accomplishments:

- Performed platform comparison analysis between GPP, FPGA, and ASICs, concluding that ASICs exhibit 1000X energy efficiency over GPPs. This contribution aids in prolonging device lifetime by ensuring energy efficient digital processing. (Chapter 2, Challenge 1)
- Designed a modified, robust synthesis flow to streamline module design in an ultra low power, batteryless BSN node. The synthesis flow resulted in all 10 test chips functioning correctly. This contribution aids in robust timing closure and standard cell functionality. (Chapter 2, Challenge 2)
- Architecture design of digital subsystem of batteryless BSN with custom MCU for control and dedicated ASIC accelerator for digital processing. Resulting architecture consumes 2.3 $\mu$ W in chip demo experiments. This contribution aids in prolonging device lifetime. (Chapter 2, Challenge 1)
- Derivation and implementation of a fast and accurate model to estimate the OCV  $\sigma/\mu$  value of logic paths operating in sub $V_T$ . The model accomplishes <11% accuracy against MC simulation. This contribution aids in correctly implementing timing closure. (Chapter 3, Challenge 4).

- Proposed and proved the importance of per path  $\sigma/\mu$  in  $\text{sub}V_T$ . The importance of the idea is exemplified by correctly identifying the critical paths in a  $\text{sub}V_T$  design, which otherwise would be missed with conventional methods. This contribution aids in correct timing closure. (Chapter 3, Challenge 4).
- Designed and incorporated into standard synthesis flow a timing method that lowers the energy overhead of robust timing closure. The two-phase clock, latch based timing method accomplishes up to 27-37% energy savings for hold time closure optimization and up to ~40% energy savings for setup time closure optimization across different yield constraints while operating at the same frequency as conventional designs. This contribution aids in the effort for robust timing closure while maintaining energy efficiency. (Chapter 4, Challenge 3)
- Identified through qualitative analysis which logic families are not suitable for use in the scope of robust, energy efficient  $\text{sub}V_T$  synthesis. This contribution serves toward accomplishing design of robust, energy efficient standard cells by pruning out logic families that have little chance of being useful in  $\text{sub}V_T$ . (Chapter 5, Challenge 5)
- Identified and designed logic families that operate robustly in  $\text{sub}V_T$  while gaining energy efficiency advantages over conventional static *CMOS* gates. Our quantitative analysis establishes a procedure to measure robustness of standard cells, and a set of experiments measure each logic family's energy efficiency. Our results show *DTCMOS* and *ONLY* logic families can improve energy efficiency by 40% and 59% respectively. The contribution fits within our goal of robust, energy efficient standard cell design. (Chapter 5, Challenge 5)

## 6.2 Contributions In Team Efforts

It should be noted that the contributions of Chapter 2 and Chapter 4 were collaborative efforts were individual efforts were combined to accomplish broader goals.



The state of the art batteryless BSN node described in Chapter 2 was accomplished through a team of 10 students between the University of Virginia and University of Washington. UVA participants on the BSN chip team consisted of Yousef Shakhsher [76], Alicia Klinefelter, Jim Boley, Aatmesh Shrivastava, and myself. The author acknowledges that separating individual contributions in a large team project can be confusing. My individual contributions on this chip are the platform comparison, the design of the modified synthesis flow, co-design of the chip architecture with Yousef Shakhsher and Helen Zhang, and co-design of the programmable FIR filter with Alicia Klinefelter. While Yousef focused on the power management and control aspects of the architecture, I focused on block implementation, hardware platform comparison and selection, synthesis flow, communication between blocks, and architecture pertaining to the interfaces with non-digital portions of the chip. Dr. Helen Zhang from the University of Washington and I together made the contribution of analog and digital integration. While I was responsible for the specifications of the FIR, Alicia Klinefelter was responsible for the RTL coding of the FIR block. The implementation was completed through my guidance of the synthesis flow.

The two-phase clock, latch based timing method described in Chapter 4 was a collaboration between the University of Virginia and nVidia. UVA participants on this project included Professor Benton Calhoun and myself. nVidia participants included Bill Dally, Tom Gray, and John Poulton. Though the entire project was done by myself, the initial idea stemmed from inspiration from both Professor Calhoun and Bill Dally. Partial completion of the project by me was done during my 2012 summer internship at nVidia.

## 6.3 Open Topics

In this section we finish this dissertation by suggesting open topics for future research originating in ideas from the projects described.

### 6.3.1 Chapter 2 (BSN Chip)

Memory leakage consumes a great portion of the digital subsystem's energy consumption, so suppressing memory leakage would be a topic of importance for future revisions of the subsystem. Many methods in the synthesis flow are made to be conservative to ensure fast design time. However, this also means modules designed in this way incur greater overheads than a more refined synthesis flow would provide. It is suggested that a full flow where more targeted modifications are made be researched to lower those overheads. The contributions of Chapter 3, 4, and 5 could be incorporated as the conclusions of those chapters do provide targeted modifications.

### 6.3.2 Chapter 3 ( $\sigma/\mu$ Estimation Method)

The drawback to a high level model that can be quickly attained is its relatively lower accuracy compared to more sophisticated methods. Thus, it is suggested to further research by closely studying the factors that contribute to a generic logic gate's  $\sigma/\mu$ , as well as outliers in  $\rho_1$  values, which we speculate are the main sources of the inaccuracy. Furthermore, the method does not cover  $\sigma/\mu$  estimation for register and latch timing metrics such as  $t_{cq}$ ,  $t_{hold}$ , and  $t_{setup}$ . Further research should be conducted to incorporate those metrics into the model.

### 6.3.3 Chapter 4 (Two-phase Latch Timing)

Since the two-phase latch timing method shows promise as an energy efficient timing scheme, it is imperative that the method can be applied to generic RTL. Thus, research should be

done to include the implementation to circuits with feedback logic. The current state of the project limits the implementation to feed forward logic only. Another main concern for this project is controlling latch overhead. Latch design that either lowers latch power or insertion delay should be researched. Lastly, alternative clock phase creation methods should be researched, since two deep level clock trees will result in increased amount of stochastic skew.

### **6.3.4 Chapter 5 (Subthreshold Standard Cells)**

Missing in our comparison analysis is the inclusion of register and latch components. Thus, research for robust, energy efficient design of registers and latches is a good starting point for future research. We should also take note that some logic families that do not fit within our scope of strictly  $\text{sub}V_T$  synthesis could become attractive if the scope were to change. Leading candidates are *bootstrap* and *array based* logic families.

# Appendix A: List of Acronyms

**AFE** Analog front end

**AFib** Atrial fibrillation

**AOCV** Advanced on-chip variation

**ASIC** Application specific integrated circuit

**BSN** Body sensor node

**CAD** Computer-aided design

**DIBL** Drain induced barrier lowering

**DLL** Delay lock loop

**DPM** Digital power manager

**DSP** Digital signal processing

**DVS** Dynamic voltage scaling

**ECG** Electrocardiogram

**ECO** Engineer exchange order

**EDA** Electronic design automation

**EEG** Electroencephalogram

**EMG** Electromyogram

**FET** Field effect transistor

**FFT** Fast Fourier transform

**FIR** Finite impulse response

**FPGA** Field programmable gate array

**GPP** General purpose processor

**MAC** Multiply accumulate

**MC** Monte Carlo

**MCU** Microcontroller

**NBTI** Negative bias temperature instability

**NV<sub>T</sub>** near-threshold

**OCV** On-chip voltage

**OSV** Output swing voltage

**PDVS** Panoptic dynamic voltage scaling

**PPA** Power, performance, and area

**PVT** Process, voltage, and temperature

**RISC** Reduced instruction set computer

**RR** R-R peak interval

**RTL** Register transfer logic

**SNM** Static noise margin

**SoC** System-on-chip

**SR** Shift register

**subV<sub>T</sub>** subthreshold

**superV<sub>T</sub>** superthreshold

**TX** Transmitter

**ULP** Ultra low power

**ULV** Ultra low voltage

**VTC** Voltage transfer characteristics

**WSN** Wireless sensor node

# Appendix B: List of Publications

- [YQ1] B. H. Calhoun, S. Khanna, **Y. Zhang**, J. Ryan, and B. Otis, "System Design Principles Combining Sub-threshold Circuits and Architectures with Energy Scavenging Mechanisms", *International Symposium on Circuits and Systems*, Paris, France, pp. 269-272, May 2010.
- [YQ2] **Y. Zhang** and B. H. Calhoun, "The Cost of Fixing Hold Time Violations in Sub-threshold Circuits", *2011 Subthreshold Microelectronics Conference*, Sept. 2011.
- [YQ3] B. H. Calhoun, .... **Y. Zhang**, et al., "Body Sensor Networks: A Holistic Approach from Silicon to Users", *IEEE Proceedings*, 2011.
- [YQ4] B. H. Calhoun, **Y. Zhang**, S. Khanna, K. Craig, Y. Shakhshier, and J. Lach, "A Sub-Threshold FPGA: Energy-Efficient Reconfigurable Logic", *GOMAC Tech*, Mar. 2011.
- [YQ5] **Y. Zhang**, Y. Shakhshier, A. T. Barth, H. Powell, S. A. Ridenour, M. A. Hanson, J. Lach, and B. H. Calhoun, "Energy Efficient Design for Body Sensor Nodes", *Journal of Low Power Electronics and Applications*, Apr. 2011.
- [YQ6] F. Zhang, **Y. Zhang** et al., "A Batteryless 19 $\mu$ W MICS/ISM-Band Energy Harvesting Body Area Sensor Node SoC", *2012 International Solid-State Circuits Conference*, Feb. 2012.
- [YQ7] Y. Shakhshier, **Y. Zhang**, B. Otis, and B. H. Calhoun, "A Custom Processor for Node and Power Management of a Battery-less Body Sensor Node in 130nm CMOS", *Custom Integrated Circuits Conference*, pp. 1-4, Sept. 2012.
- [YQ8] A. Klinefelter, **Y. Zhang**, B. Otis, and B. H. Calhoun, "A Programmable 34 nW/Channel Sub-Threshold Signal Band Power Extractor on a Body Sensor Node SoC", *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 59, issue 12, pp. 941, Dec. 2012.
- [YQ9] **Y. Zhang**, F. Zhang, Y. Shakhshier, J. D. Silver, A. Klinefelter, M. Nagaraju, J. Boley, J. Pandey, A. Shrivastava, E. J. Carlson, et al., "A Batteryless 19  $\mu$ W MICS/ISM-Band Energy Harvesting Body Sensor Node SoC for ExG Applications", *Journal of Solid State Circuits*, vol. 48, issue 1, pp. 199-213, Jan. 2013.
- [YQ10] N. Gilbert, **Y. Zhang**, J. Dinh, B. Calhoun, and S. Hollmer, "A 0.6V 8 pJ/write Non-Volatile CBRAM Macro Embedded in a Body Sensor Node for Ultra Low Energy Applications", *Symposium on VLSI Circuits*, 2013.
- [YQ11] **Y. Zhang** and B. H. Calhoun, "Hold Time Closure for Subthreshold Circuits Using a Two-Phase, Latch Based Timing Method", *S3S Conference*, Oct. 2013.
- [YQ12] **Y. Zhang** and B. H. Calhoun, "Fast, Accurate Variation-Aware Path Timing Computation for Sub-threshold Circuits", *ISQED* 2014.

# Bibliography

- [1] G. Z. Yang, Ed. “Body Sensor Networks”, Springer-Verlag, London, UK, 2006.
- [2] A. T. Barth, M. A. Hanson, H. C. Powell, and J. Lach, “TEMPO 3.1: A Body Area Sensor Network Platform for Continuous Movement Assessment”, *Proceedings of the 2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks*, pp. 71–76, Berkeley, CA, USA, 3–5 June 2009.
- [3] B. H. Calhoun, A. Wang, A. Chandrakasan, “Modeling and Sizing for Minimum Energy Operation in Sub-threshold Circuits”, *IEEE Journal of Solid-State Circuits*, 2005.
- [4] <http://techcrunch.com/tag/internet-of-things/>
- [5] M.H. Abu-Rahma and M. Anis, “A Statistical Design-Oriented Delay Variation Model Accounting for Within-Die Variations”, *Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, issue 11, pp. 1983-1995, Nov. 2008.
- [6] M.J.M. Pelgrom, A.C.J. Duinmaijer, and A.P.G. Welbers, “Matching Properties of MOS Transistors”, *Journal of Solid-State Circuits*, vol. 24, issue 5, pp. 1433-1439, Oct. 1989.
- [7] A. Dasdan, S. Kolay, and M. Yazgan, “Derating for Static Timing Analysis: Theory and Practice”, *Quality of Electronic Design*, pp. 719-727, March 16-18, 2009.
- [8] J. Kwong and A. Chandrakasan, “Variation-driven Device Sizing for Minimum Energy Sub-threshold Circuits”, *Proceedings of the 2006 International Symposium on Low Power Electronics and Designs*, pp. 8-13, 2006.
- [9] Y. Zhang, F. Zhang, et. al, “A Batteryless 19 $\mu$ W MICS/ISM-Band Energy Harvesting Body Sensor Node SoC for ExG Applications”, *Journal of Solid State Circuits*, vol. 48, issue 1, pp. 199-213, Jan. 2013.
- [10] S. Jocke, J. Bolus, S. N. Wooters, A. D. Jurik, A. C. Weaver, T. N. Blalock, and B. H. Calhoun, “A 2.6- $\mu$ W Sub-threshold Mixed-signal ECG SoC”, *Proceedings of the Symposium on VLSI Circuits 2009*, pp. 60-61, Kyoto, Japan, 16–18 June 2009.
- [11] B. Zhai, L. Nazhandali, J. Olson, A. Reeves, M. Minuth, R. Helfand, S. Pant, D. Blaauw, and T. Austin, “A 2.60 pJ/inst Subthreshold Sensor Processor for Optimal Energy Efficiency”, *Proceedings of the Symposium on VLSI Technology and Circuits*, pp. 154-155, Hawaii, HI, USA, June 2006.
- [12] J. Kwong, Y. Ramadass, N. Verma, and A. Chandrakasan, “A 65 nm Sub-V<sub>t</sub> Microcontroller with Integrated SRAM and Switched Capacitor DC-DC Converter”, *IEEE Journal of Solid-State Circuits*, vol. 44, pp. 115–126, 2009.
- [13] A. Wang, and A. Chandrakasan, “A 180mV FFT Processor Using Subthreshold Circuit Techniques”, *Proceedings of the IEEE International Solid-State Circuits Conference*, pp. 292-293, San Francisco, CA, USA, Feb. 6-10 2005.
- [14] Y. Pu, J. P. de Gyvez, H. Corporaal, Y. Ha, “An Ultra-low-energy/frame Multi-standard JPEG Co-processor in 65nm CMOS with Sub/near-threshold Power Supply”, *Proceedings of the IEEE International Solid-State Circuits Conference*, pp. 146–147, San Francisco, CA, USA, Feb. 8-12 2009.
- [15] J. F. Ryan and B. H. Calhoun, “A Sub-threshold FPGA with Low-swing Dual-VDD Interconnect in 90nm CMOS”, *Proceedings of the Custom Integrated Circuits Conference*, San Jose, CA, USA, Sept. 18-21 2010.

- [16] P. Meinerzhagen, O. Andersson, Y. Sherazi, A. Burg, and J. Rodrigues, "Synthesis Strategies for Sub- $V_T$  Systems", *20<sup>th</sup> European Conference on Circuit Theory and Design*, pp. 552-555, Aug. 2011.
- [17] H. Kim, R. F. Yazicioglu, S. Kim, N. Van Helleputte, A. Artes, M. Konijnenburg, J. Huysen, J. Penders, and C. Van Hoof, "A Configurable and Low-Power Mixed Signal SoC for Portable ECG Monitoring Applications," *2011 Symposium on VLSI Circuits*, pp. 142-143, June 2011.
- [18] N. Verma, A. Shoeb, J. Bohorquez, J. Dawson, J. Guttag, and A. P. Chandrakasan, "A Micro-Power EEG Acquisition SoC With Integrated Feature Extraction Processor for a Chronic Seizure Detection System," *IEEE Journal of Solid-State Circuits*, vol.45, no.4, pp. 804-816, April 2010.
- [19] G. Chen, M. Fojtik, K. Daeyeon, D. Fick, J. Park. M. Seok, M.-T. Chen, Z. Foo, D. Sylvester, and D. Blaauw, "Millimeter-Scale Nearly Perpetual Sensor System with Stacked Battery and Solar Cells," *2010 IEEE Solid-State Circuits Conference Digest of Technical Papers*, pp. 288-289, Feb 2010.
- [20] S. Rai, J. Holleman, J. N. Pandey, F. Zhang, and B. Otis, "A 500 $\mu$ W Neural Tag with 2 $\mu$ Vrms AFE and Frequency-Multiplying MICS/ISM FSK Transmitter," *2009 IEEE Solid-State Circuits Conference Digest of Technical Papers*, pp. 212-213, 213a, Feb. 2009.
- [21] Y. Long, J. Bae, S. Lee, T. Roh, K. Song, and H.-J. Yoo, "A 3.9 mW 25-Electrode Reconfigured Sensor for Wearable Cardiac Monitoring System," *IEEE Journal of Solid-State Circuits*, vol. 46, no. 1, pp. 353-364, Jan. 2011.
- [22] J. M. Rabaey, A. Abnous, Y. Ichikawa, K. Seno, M. Wan, "Heterogeneous Reconfigurable Systems", *Proceedings of the IEEE Workshop on Signal Processing Systems, SiPS 97 Design and Implementation formerly VLSI Signal Processing*, pp. 24-34, Leicester, UK, Nov. 1997.
- [23] H. Zhang, V. Prabhu, V. George, M. Wan, M. Benes, A. Abnous, J. M. Rabaey, "A 1-V Heterogeneous Reconfigurable DSP IC for Wireless Baseband Digital Signal Processing", *IEEE J. Solid-State Circuits*, pp. 1697-1704, 2000.
- [24] D. Lake and J.R. Moorman, "Accurate Estimation of Entropy in Very Short Physiological Time Series: The Problem of Atrial Fibrillation Detection in Implanted Ventricular Devices", *American Journal of Physiology Heart and Circulatory Physiology*, Jan 2011.
- [25] Y. Zhang, et al., "A Batteryless 19  $\mu$ W MICS/ISM-Band Energy Harvesting Body Sensor Node SoC for ExG Applications", *Journal of Solid-State Circuits*, vol. 48, issue 1, pp. 199-213, 01/2013.
- [26] <http://www.microchip.com/>
- [27] K. Craig, Y. Shakhsher, and B. H. Calhoun, "Optimal Power Switch Design for Dynamic Voltage Scaling from High Performance to Subthreshold Operation", *International Symposium on Low Power Electronics Design*, 2012.
- [28] E. Brunvand, "Digital VLSI Chip Design with Cadence and Synopsys CAD Tools", *Addison-Wesley*, 2010.
- [29] Y. Shakhsher, Y. Zhang, B. Otis, and B. H. Calhoun, "A Custom Processor for Node and Power Management of a Battery-less Body Sensor Node in 130nm CMOS", *Custom Integrated Circuits Conference*, San Jose, 09/2012.
- [30] <http://ecg.mit.edu/>



- [31] A. Klinefelter, Y. Zhang, B. Otis, and B. H. Calhoun, "A Programmable 34 nW/Channel Sub-Threshold Signal Band Power Extractor on a Body Sensor Node SoC", *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 59, issue 12, pp. 941, 12/2012.
- [32] J. Pan and W. J. Tompkins, "A Real-Time QRS Detection Algorithm," *IEEE Transactions on Biomedical Engineering*, vol. BME-32, issue 3, pp. 230-236, March 1985.
- [33] Y. Zhang and B. H. Calhoun, "Fast, Accurate Variation-Aware Path Timing Computation for Sub-threshold Circuits", *ISQED* 2014.
- [34] B. Lasbouygues, R. Wilson, N. Azemard, and P. Maurine, "Timing Analysis in Presence of Supply Voltage and Temperature Variations", *Proceedings of the 2006 International Symposium on Physical Design*, pp. 10-16, 2006.
- [35] S. Bhattacharya, <http://www.edn.com/design/integrated-circuit-design/4363478/Path-specific-derating-to-reduce-timing-pessimism-a-proposal>, 2010.
- [36] <http://www.techdesignforums.com/practice/guides/on-chip-variation-ocv/>
- [37] "Advanced On-chip-variation Timing Analysis for Nanometer Designs", *Incentia Design Systems, Inc. Technical Document*, [http://www.incentia.com/products/TimeCraft\\_01.pdf](http://www.incentia.com/products/TimeCraft_01.pdf).
- [38] R. Rith, J. Gu, A. Wang, S. Datla, G. Gammie, D. Buss, and A. Chandrakasan, "Non-linear Operating Point Statistical Analysis for Local Variations in Logic Timing at Low Voltage", *Design, Automation and Test in Europe*, pp. 965-968, March 2010.
- [39] A. Agarwal, D. Blaauw, V. Zolotov, S. Sundareswaran, M. Zhao, K. Gala, and R. Panda, "Statistical Delay Computation Considering Spatial Correlations", *Design Automation Conference*, pp. 271-276, Jan. 2003.
- [40] M.H. Abu-Rahma and M. Anis, "A Statistical Design-Oriented Delay Variation Model Accounting for Within-Die Variations", *Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, issue 11, pp. 1983-1995, Nov. 2008.
- [41] H. Mahmoodi, S. Mukhopadhyay, and K. Roy, "Estimation of Delay Variations Due to Random-Dopant Fluctuations in Nanoscale CMOS Circuits", *Journal of Solid-State Circuits*, vol. 40, issue 9, pp. 1787-1796, Sept. 2005.
- [42] Y. Cao and L. T. Clark, "Mapping Statistical Process Variations Toward Circuit Performance Variability: An Analytical Modeling Approach", *Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, issue 10, pp. 1866-1873, Oct. 2007.
- [43] E. L. Crow and K. Shimizu, "Lognormal Distributions: Theory and Applications", *Marcel Dekker*: New York, NY, USA 1988; pp. 195–210.
- [44] J. Keane, E. Hanyong, K. Tae-Hyoung, S. Sapatnekar, and C. Kim, "Subthreshold Logical Effort: A Systematic Framework for Optimal Subthreshold Device Sizing", *Proceedings of the 43rd Design Automation Conference*, pp. 425–428, June 24, 2006.
- [45] Y. Zhang and B. H. Calhoun, "Hold Time Closure for Subthreshold Circuits Using a Two-Phase, Latch Based Timing Method", *S3S Conference*, Oct. 2013.
- [46] M. Wieckowski, et al., "Timing Yield Enhancement Through Soft Edge Flip-Flop Based Design," *Custom Integrated Circuits Conference*, Sept. 2008.
- [47] M. Wieckowski, M. P. Young, C. Tokunaga, D. W. Kim, Z. Foo, D. Sylvester, and D. Blaauw, "Timing Yield Enhancement Through Soft Edge Flip-Flop Based Design", *Custom Integrated Circuits Conference*, pp. 543-546, Sept. 2008.

- [48] E. Laulainen, M. J. Turnquist, J. Makipaa, and L. Koskinen, "Adaptive Subthreshold Timing-error Detection 8 bit Microcontroller in 65 nm CMOS", *2012 IEEE International Symposium on Circuits and Systems*, pp. 2953-2956, May 2012.
- [49] D. Ernst, N. S. Kim, S. Das, S. Pant, T. Pham, R. Rao, C. Ziesler, D. Blaauw, T. Austin, and T. Mudge, "Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation", *ACM/IEEE International Symposium on Microarchitecture*, pp. 7-18, Dec. 2003.
- [50] D. Blaauw, S. Kalaiselvan, K. Lai, W.-H. Ma, S. Pant, C. Tokunaga, S. Das, and D. Bull, "RazorII: In-Situ Error Detection and Correction for PVT and SER tolerance", *IEEE International Solid-State Circuits Conference*, Feb. 2008.
- [51] M. Fojtik, D. Fick, Y. Kim, N. Pinckney, D. Harris, D. Blaauw, and D. Sylvester, "Bubble Razor: An Architecture-Independent Approach to Timing-Error Detection and Correction", *IEEE International Solid-State Circuits Conference*, Feb. 2012.
- [52] F. Hsu, P. Solecky, and L. Zobniw, "Selective Controllability: A Proposal for Testing and Diagnosis", *15<sup>th</sup> Conference on Design Automation*, pp. 110-116, June 1978.
- [53] V. Chandramouli, N. Michell, and K. F. Smith, "A New, Precharged, Low-Power Logic Family for GaAs Circuits", *Journal of Solid-State Circuits*, pp. 140-143, Feb. 1995.
- [54] R. B. Wunderlich, B. P. Degnan, and P. Hasler, "Capacitively-Biased Floating-Gate CMOS: a New Logic Family", *International Symposium on Circuits and Systems*, pp. 3728-3731, May 2007.
- [55] O. Takahashi, N. Aoki, J. Silbermah, and S. Dhong, "1 GHz Logic Circuits with Sense Amplifiers", *Symposium on VLSI Circuits*, pp. 110-111, June 1998.
- [56] H. Yamaoka, M. Ikeda, and K. Asada, "A High-Speed Logic Circuit Family with Interdigitated Array Structure for Deep Sub-micron IC Design", *Proceedings of the 29<sup>th</sup> European Solid-State Circuits Conference*, pp. 189-192, Sept. 2003.
- [57] A. G. Dickinson and J. S. Denker, "Adiabatic Dynamic Logic", *Proceedings of the IEEE Custom Integrated Circuits Conference*, pp. 282-285, May 1994.
- [58] S. Kim and M. C. Papaefthymiou, "True Single-Phase Energy-Recovering Logic for Low-Power, High-Speed VLSI", *Proceedings of the International Symposium on Low Power Electronics and Design*, pp. 167-172, Aug. 1998.
- [59] S. Y. Choe and G. A. Rigby, "A 1V Bootstrapped CMOS Digital Logic Family", *Proceedings of the 23<sup>rd</sup> European Solid-State Circuits Conference*, pp. 352-355, Sept. 1997.
- [60] B.-S. Kong, D.-O. Kang, and Y.-H. Jun, "A Bootstrapped CMOS Circuit Technique for Low-Voltage Application", *6<sup>th</sup> International Conference on VLSI and CAD*, pp. 289-292, Oct. 1999.
- [61] Y.-K. Tseng and C.-Y. Wu, "A New True-Single-Phase-Clocking BiCMOS Dynamic Pipelined Logic Family for High-Speed, Low-Voltage Pipelined System Applications", *Journal of Solid-State Circuits*, pp. 68-79, Jan. 1999.
- [62] A. M. Fahim and M. I. Elmasry, "SC2L: A Low-Power High-Performance Dynamic Differential Logic Family", *Proceedings of the 1999 International Symposium on Low Power Electronics and Design*, pp. 88-90, 1999.
- [63] R. J.-H. Sung and D. G. Elliott, "Clock-Logic Domino Circuits for High-Speed and Energy-Efficient Microprocessor Pipelines", *IEEE Transactions on Circuits and Systems II*, pp. 460-464, vol. 54, issue 5, May 2007.

- [64] L. Heller, W. Griffin, J. Davis, and N. Thoma, "Cascode Voltage Switch Logic: A Differential CMOS Logic Family", *International Solid-State Circuits Conference*, pp. 16-17, Feb. 1984.
- [65] V. Majidzadeh, S. M. Alavi, and A. Afzali-Kusha, "Design of Merged Differential Cascode Voltage Switch with Pass-gate (MDCVSPG) Logic for High-Performance Digital Systems", *17<sup>th</sup> International Conference on Microelectronics*, Dec. 2005.
- [66] D. W. Kang and Y.-B. Kim, "Design of Enhanced Differential Cascode Voltage Switch Logic (EDCVSL) Circuits for High Fan-in Gate", *15<sup>th</sup> Annual IEEE International ASIC/SoC Conference*, pp. 309-313, Sept. 2002.
- [67] E. Albuquerque and M. Silva, "A New Low-Noise Logic Family for Mixed-Signal Integrated Circuits", *IEEE Transactions on Circuits and Systems I*, pp. 1498-1500, vol. 46, issue 12, Dec. 1999.
- [68] J. Nyathi and B. Bero, "Logic Circuits Operating in Subthreshold Voltages", *Proceedings of the 2006 International Symposium on Low Power Electronics and Design*, pp. 131-134, Oct. 2006.
- [69] H. Soeleman, K. Roy, and B. C. Paul, "Robust Subthreshold Logic for Ultra-Low Power Operation", *IEEE Transactions on VLSI Systems*, pp. 90-99, vol. 9, issue 1, Feb. 2001.
- [70] A. N. Bhoj and N. K. Jha, "Design of Logic Gates and Flip-Flops in High-Performance FinFET Technology", *IEEE Transactions on VLSI Systems*, pp. 1975-1988, vol. 21, issue 11, Nov. 2013.
- [71] A. Kaizerman, S. Fisher, and A. Fish, "Subthreshold Dual Mode Logic", *IEEE Transactions on VLSI Systems*, pp. 979-983, vol. 21, issue 5, May 2013.
- [72] N. Reynders and W. Dehaene, "Variation-Resilient Sub-threshold Circuit Solutions for Ultra-Low-Power Digital Signal Processors with 10MHz Clock Frequency", *2012 Proceedings of the ESSCIRC*, pp. 474-477, Sept. 2012.
- [73] B. H. Calhoun, F. A. Honore, and A. P. Chandrasakan, "A Leakage Reduction Methodology for Distributed MTCMOS", *Journal of Solid-State Circuits*, pp. 818-826, vol. 39, issue 5, May 2004.
- [74] J. Rabaey, A. Chandrakasan, and B. Nikolic, "Digital Integrated Circuits (2<sup>nd</sup> Edition)", *Prentice Hall Electronics and VLSI Series*, Upper Saddle River, NJ: Pearson Education, 2003.
- [75] <http://www.anandtech.com/show/5555/intel-at-isscc-12-more-research-into-near-threshold-voltage>
- [76] Y. Shaksheer, "Lifetime Improvements in Body Sensor Networks", PhD Dissertation, *University of Virginia Library Archives*, 2013.