UNIVERSITY OF CALIFORNIA
SANTA CRUZ

REDUCING VARIABILITY IN SUBTHRESHOLD CIRCUITS

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER ENGINEERING

by

Rajsaktish Sankaranarayanan

March 2017

The Dissertation of Rajsaktish Sankaranarayanan
is approved:

Professor Matthew Guthaus, Chair

Professor Jose Renau

Professor Jishen Zhao

Tyrus Miller
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

List of Tables

Abstract

Reducing Variability in Subthreshold Circuits

by

Rajsaktish Sankaranarayanan

Reduced form-factor in portable electronics has made energy-efficiency the primary target. Subthreshold operation is a technique delivering orders of magnitude greater energy-savings compared to other techniques. However, circuits in subthreshold are very sensitive to the impact of process variation. Utilizing subthreshold operation and leveraging energy-efficiency necessitates compensation techniques to mitigate process variation.

The specific contributions of this thesis are: design and implementation of a subthreshold FPGA chip using body bias as a compensation mechanism against threshold voltage variation. Analysis of performance and energy of a subthreshold FPGA using a characterization framework. Using this framework, the minimum energy point of the subthreshold FPGA was found to be in deep subthreshold, and a performance window of 30x with a 2x energy range was identified. A design methodology to mitigate threshold voltage variation by delivering optimized and adaptive body bias in circuits with standard cells is proposed. Using two algorithms, this methodology produces standby energy savings of up to 21.06% on average and active energy savings of up to 18.84% on average.

Acknowledgments

Returning to graduate school to pursue my PhD has been a very enriching experience for me. In this process, I have interacted with and learnt from many people, for which I am grateful. There are a few people who have played a significant part in my development, whom I want to thank.

My advisor Professor Matthew Guthaus, for giving me time and space to grow as a student and researcher. For this I am indebted to him. His clarity of ideas, patience, insightful feedback are personal attributes I have greatly benefited from. His deep grasp of many subject areas and interpersonal management are truly admirable.

I am very thankful to Professor Jose Renau and Professor Jishen Zhao for agreeing to be on my reading committee and for the interactions I have had with them over the years.

Navigating administrative and immigration processes would have been daunting for me, were it not for the generous help from Emily Gregg in the Department of Computer Engineering and Adrienne Bergenfeld in the International Student and Scholar Services.

The National Science Foundation (NSF), for the grant which supported a part of this research.

In the VLSIDA Lab, I have enjoyed working with several colleagues. Keven Woo, Andrew Hill, Seokjoong Kim, Xuchu Hu, Walter Condley, Sheldon Logan, Benjamin Lacara, Jeffrey Butera, Brian Chen, Riadul Islam, Bin Wu, Hany Fahmy, Ping-

Yao Lin and Rebecca Rashkin - I enjoyed the knowledgeable discussions and lively chats over the years with all of you.

During my two internship opportunities with Intel, Santa Clara, I greatly benefited working and learning from my manager Nagib Hakim and team members Amitava Bhaduri and Kalyan Donepudi. I am very thankful to them.

In the past few months, the encouragement and support from my manager Shaoyi Wang and team members at Intel, San Jose helped me balance my job and writing this dissertation. I am very thankful to them.

These past few years, although I have not spent much time with my friends, I am thankful for their encouragement. Karthikeyan Ramasamy, Prabhu Patil, Ram Gooty, Srinivas Yelisetti, Vivek Raghuraman, Sangeetha Vasu, Karthikeyan Subramanian, Sandhya Srinivasan - you have all been very understanding.

My cousin Shiv and family, for the cheer and support during my graduate school years.

My parents-in-law, for their loving patience all along and generous help during their visits.

My elder brother Swaroop and family, for their unstinting support and encouragement all these years. You made the passage of time feel briefer.

I am very grateful to my parents, for their encouragement, loving support and prayers for my growth. You have been a tremendous source of strength for me always.

My son Vidyuth, who was born a few months ago, has been a source of joy for me in completing this dissertation.

My wife Ramya, for being there for me continually. Her love, support and encouragement energized me to view each day with fresh spirit and move forward. To her, with all my dedication.

Chapter 1

Introduction

Portable electronics has created a strong demand for energy efficient circuit design. Mobile devices are extensively used for long periods due to their functionality and convenience. With reduced form factor in these devices, their overall operation is now limited by battery life. This places tight requirements on the hardware design. Technology roadmaps of the previous decades centered on delivering increasing performance. Future growth of electronics is dependent on the industry being able to deliver innovative solutions keeping power and energy as the defining constraints.

Power saving techniques are often used in circuit design to save active power, standby power or both. Some of the commonly used techniques include: supply voltage and threshold voltage scaling [20], dynamic voltage and frequency scaling [46], multi-threshold devices [40, 1], dynamic threshold voltage [30], gate sizing [31], clock gating [35] and transistor reordering. Among these techniques supply voltage scaling offers maximal savings as it reduces both dynamic and standby power. Dynamic power has

a squared dependence, while standby power has an exponential dependence on supply voltage. This enables voltage scaling to save power in both modes of circuit operation. Operating a circuit using a supply voltage less than the threshold voltage is known as Subthreshold Operation.

Designing circuits for subthreshold operation is challenging due to the reduced operating voltages. At low voltages circuits are more sensitive to the impact of process variation. As a result, even small variation can lead to significant change in performance, which in turn impacts circuit energy efficiency. To continue subthreshold operation there is a pressing need to evaluate the constituent factors in energy efficiency, identify challenges in circuit design and develop solutions to address them.

This thesis studies the performance of circuits in subthreshold to determine the functional components contributing to circuit energy and delay. Circuit sensitivity to process variation is evaluated and a methodology to reduce worst-case variation in energy and performance is proposed.

The specific contributions are:

- design and implementation of a subthreshold Field Programmable Gate Array (FPGA) chip.
- analysis of energy, performance and block level contribution of a subthreshold FPGA using a high level characterization framework [48].
- development of a variation model and application to benchmarks to measure the impact of variation in subthreshold.

- development of a methodology to mitigate variation in subthreshold [49].

The rest of this thesis is organized as follows: Chapter 2 provides background material on subthreshold circuit operation, variability in subthreshold and FPGA design methodology. Chapter 3 presents the architecture, design and implementation of a Single V_{dd} Subthreshold FPGA chip. A methodology to characterize the power and performance of an FPGA using block level analysis is discussed in Chapter 4. Chapter 5 presents sensitivity of subthreshold circuits to impact of process variation and a methodology to mitigate variation. Conclusions and potential future work are discussed in Chapter 6.

Chapter 2

Background

Subthreshold operation has gained acceptance for digital and analog computation [6],[18]. In subthreshold, the relationship between various transistor parameters is different from that of nominal supply voltage operation. This necessitates understanding the mechanism of subthreshold operation before moving to application specific discussions. This chapter discusses the fundamentals of subthreshold operation, and the major challenge in designing for subthreshold, namely, impact of process variation on circuit performance. This is followed by an overview of Field Programmable Gate Arrays (FPGAs), and development of FPGAs targeting subthreshold operation.

2.1 Subthreshold operation

Digital circuit operation using CMOS technology typically assumes an ideal model of transistor switching operation. This is demonstrated by the I-V model in which transistor current flows between the drain and source terminals of the transistor when the

applied gate voltage (V_{gs}) exceeds the threshold voltage (V_{th}) of the transistor. In practice, the current does not abruptly stop when the applied gate voltage falls below the threshold voltage. Instead it follows an exponential decrease [3] given by:

$$I_{sub} = A \cdot e^{\frac{\Delta Vth}{\eta V_T}} \cdot e^{(\frac{V_{gs} - Vtho - \gamma V_s + \eta V_{ds}}{nV_T})} \cdot (1 - e^{\frac{-V_{ds}}{V_T}}) \tag{2.1}$$

where $A = \mu_0 \cdot C_{ox} \cdot \frac{W}{L_{eff}} \cdot {V_T}^2 \cdot e^{1.8} \cdot e^{\frac{\Delta Vth}{\eta V_T}}$, μ_0 is the carrier mobility, C_{ox} is the gate-oxide capacitance, W is the transistor width, L_{eff} is the effective channel length, η is the Drain-Induced Barrier Lowering (DIBL) coefficient, γ is the Body effect coefficient, V_{tho} is the Zero bias threshold voltage, V_T is the Thermal voltage, n is the Subthreshold swing coefficient, ΔV_{th} is the transistor-to-transistor threshold voltage variation, V_{gs} is the voltage between gate and source terminals and V_{ds} is the voltage between drain and source terminals.

To obtain a first-order approximation, certain transistor effects can be ignored, including DIBL, V_{ds} roll-off and threshold voltage variation. Assuming transistor body and source terminals are at the same potential and rail-to-rail output swing ($V_{ds} = V_{dd}$), Equation 2.1 can be approximated as

$$I_{sub} = I_0 \cdot e^{\frac{V_{dd} - V_{th0}}{nV_T}} \tag{2.2}$$

where $I_0 = \mu_0 \cdot C_{ox} \cdot \frac{W}{L_{eff}} \cdot {V_T}^2$ is the drain current when $V_{gs} = V_{th0}$ and is dependent on process parameters. In the remainder of this section, all references to threshold voltage (V_{th}) indicate the zero-bias threshold voltage, unless indicated otherwise.

Since V_{dd} is an exponential term, lowering it results in an exponential current reduction in the subthreshold region. Figure 2.1 shows exponential dependence of drain

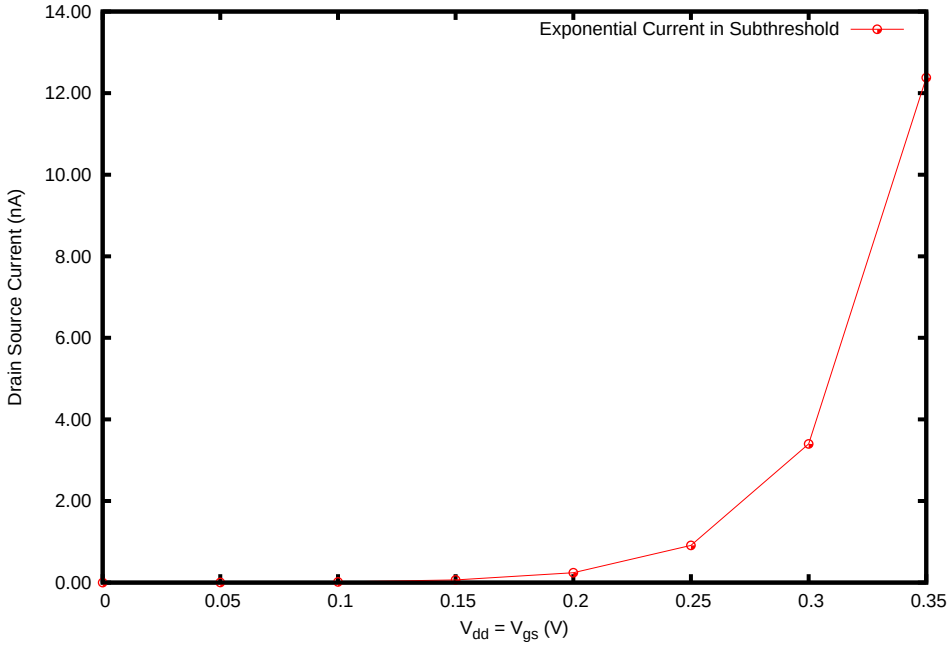

Figure 2.1: Drain Source current varies exponentially with the applied voltage

current on gate voltage, in an NMOS transistor using a TSMC-180nm process.

2.1.1 Subthreshold Performance

Circuit performance is determined by a number of factors including threshold voltage, load capacitance and supply voltage. Circuit switching is performed by the charging and discharging of load capacitance by the transistor drain current. The drain current of a transistor in saturation [55] is given by:

$$I_{ds:sat} = \frac{\mu \cdot C_{ox}}{2} \cdot \frac{W}{L} \cdot (V_{gs} - V_{th})^2 \cdot (1 + \lambda \cdot V_{ds}) \tag{2.3}$$

where μ is the mobility of transistor majority carrier, C_{ox} is the gate-oxide thickness, W is the transistor width, L is the transistor length, V_{gs} is the voltage between gate and

source terminals of transistor, V_{ds} is the voltage between drain and source terminals of transistor, V_{th} is the transistor threshold voltage and λ is the transistor channel length modulation coefficient.

In a typical inverter with NMOS and PMOS devices sized for equal rise and fall times, the propagation delay is the average of rise and fall time transitions approximated by:

$$t_{pd} = \frac{C_L \cdot V_{dd}}{I_{ds:sat}} \tag{2.4}$$

where C_L is the load capacitance, V_{dd} is the supply voltage, and $I_{ds:sat}$ is the saturation current of transistor. The influence of some of these parameters in subthreshold is different compared to peak voltage operation. In peak voltage operation, the switching capacitance is dominated by the gate oxide capacitance at the load and is independent of the applied voltage. In subthreshold, in addition to gate oxide capacitance voltage dependent capacitances namely, fringe, overlap and depletion capacitances also contribute to loading. The load capacitance is charged using the transistor leakage current given by Equation 2.1. The propagation delay of a typical inverter in subthreshold can be approximated as follows:

$$t_{pd} = \left(\frac{C_{tot} \cdot V_{dd}}{I_{sub}} \right) \tag{2.5}$$

Substituting 2.2 in 2.5 gives,

$$t_{pd} = \left(\frac{C_{tot} \cdot V_{dd}}{I_0 \cdot exp^{\left(\frac{V_{dd}-V_{th}}{n \cdot V_T}\right)}} \right) \tag{2.6}$$

where C_{tot} is the total capacitance taking into account voltage dependent capacitances and I_0 is the transistor drain current when $V_{gs} = V_{th}$

From Equation 2.6 it can be seen, the delay is exponentially dependent on the applied voltage. Circuits with stringent energy constraints and low performance requirements can benefit from this, by trading performance to save energy.

Analog circuits with very low current requirements were some of the earliest applications for subthreshold operation. MOS transistors in conjunction with bipolar transistors were used in developing an on-chip voltage reference [64]. Since the transistors operate in weak inversion, the voltage reference is temperature independent. A very low power quartz oscillator operating at 32 KHz and consuming 0.1 μ Watt was implemented using transistors in weak inversion [19].

The smallest supply voltage at which a digital inverter was functional was found to be 200 mV [50]. This supply voltage was a function of device thermal voltage (T) and majority carrier charge (q) and expressed as $8kT/q$ where k = Boltzmann constant. The performance of logic gates in subthreshold was compared using pseudo-NMOS and CMOS implementations. Gates without PMOS transistor stacks performed better in pseudo-NMOS implementation [22]. Digital circuits with very low performance requirements like hearing-aid applications were able to take advantage of subthreshold operation. A 22 KHz adaptive filter [23] was designed to operate at a supply voltage of 400 mV.

While these applications helped establish subthreshold operation as a viable technique, the focus was on leveraging the underlying process technology to achieve

relaxed performance targets.

2.1.2 Total Energy Reduction

In CMOS circuits there are three sources of power consumption. The first, switching power, is consumed by the charging and discharging of load capacitances, due to signal transitions. The second, short-circuit power is consumed due to the current flowing through the transistors when a path between V_{dd} and *ground* is formed momentarily during switching. These two components are together referred to as dynamic power. The third, static power is consumed when the circuit is in standby and not switching.

Assuming periodic input and output waveforms, the instantaneous power consumption of a transistor at any time instant (t) is given by

$$P(t) = i(t) \cdot V_{dd} \tag{2.7}$$

and the energy consumed is the instantaneous power over a period of time (T) given by

$$E = \int_0^T P(t)\,\mathrm{d}t. \tag{2.8}$$

The average power drawn by a transistor in the time T is given by

$$P_{avg} = \frac{E}{T} \tag{2.9}$$

which includes the dynamic and static components of power.

Dynamic power depends on the load capacitance (C), supply voltage (V_{dd}), toggle rate of the circuit nodes (α) and frequency of operation (f) given by

$$P_{dyn} = \alpha \cdot C \cdot V_{dd}^2 \cdot f. \tag{2.10}$$

Assuming the circuit is operated at the maximum designed frequency (f) such that

$$T = \frac{1}{f} \tag{2.11}$$

switching energy for one cycle of operation ($\alpha = 1$) becomes

$$E_{dyn} = C \cdot V_{dd}^2. \tag{2.12}$$

Static power on the other hand, is the product of the leakage current of the transistors, when they are off, and the supply voltage,

$$P_{static} = I_{sub} \cdot V_{dd}, \tag{2.13}$$

using which the leakage energy during one cycle of operation can be estimated as,

$$E_{leak} = I_{sub} \cdot V_{dd} \cdot T. \tag{2.14}$$

Using Equation 2.2 this can be rewritten as,

$$E_{leak} = I_0 \cdot e^{\frac{V_{dd} - V_{th0}}{nV_T}} \cdot V_{dd} \cdot T. \tag{2.15}$$

From this, an expression for the total energy, in terms of the dynamic and static energy components can be written as,

$$E_{total} = \left(C \cdot V_{dd}^2 + I_0 \cdot V_{dd} \cdot e^{\frac{V_{dd} - V_{th0}}{nV_T}} \right). \tag{2.16}$$

From this, it can be seen energy efficiency and total energy minimization requires a balance between dynamic and leakage energy components.

Expressions similar to Equation 2.16 have been proposed [11], [13]. Impact of transistor size on overall energy was analyzed and found that minimum sized devices

were optimal to reduce energy [11]. The optimal supply voltage at which minimum energy consumption occurs was determined to be in subthreshold [13].

Many other subthreshold applications operating at the minimum energy point, and consuming very low energy per computation were developed. Using a Fast Fourier Transform (FFT) processor [8] with variable bit precision, minimum energy analysis showed that the optimal power supply voltage occured in subthreshold. This FFT processor could operate down to 180 mV and consumed minimum energy of 155 nJ/FFT operating at a frequency of 10 KHz using a supply voltage of 350 mV. An 8-bit subthreshold processor [52] operating at a target low frequency range consumed 3.5 pJ/instruction at 350 mV and functional down to 200 mV. Using supply voltage scaling over the target frequency range, the minimum energy point occured at a target frequency of 80 KHz. Device optimization at the transistor level targeting minimum energy in subthreshold operation [6] was studied. Reducing junction capacitance by eliminating halo doping, leading to lower energy consumption was considered. However, this requires process technology development specifically targeting subthreshold operation. A scaling strategy [53] for transistors capable of achieving optimal subthreshold performance and energy was proposed. Also, this strategy improved static noise margin (SNM) of SRAM cells. A 256 Kb SRAM with read and write functionality at 350 mV using a modified bit-cell architecture was proposed. The write functionality was accomplished using write-assist circuitry [42]. Standby-V_{dd} in subthreshold memory was optimized to reduce leakage, by using row/column redundancy without sacrificing yield [56]. These circuit applications helped establish subthreshold operation as a low power solution, and

brought attention to the major challenge in the design of subthreshold circuits, namely impact of process variation on circuit performance and energy.

2.2 Variability in subthreshold

Circuit design for subthreshold operation is challenging due to greater sensitivity of transistors to process variation at reduced supply voltages. Even small variation in parameters like threshold voltage, gate-oxide thickness and channel length can lead to a large variation in circuit performance, potentially causing a timing failure. Overcoming this requires one or more measures like design guardbanding, raising the supply voltage or reducing operating frequency any of which will impact overall energy consumption. So, to mitigate variation in subthreshold and retain energy efficiency, optimal compensation techniques are required.

2.2.1 Mitigation using design techniques

Circuits operating at peak supply voltage are sensitive to variation in device geometry, doping fluctuation, interconnect variation and operating temperature. At low target frequencies, circuit operating temperatures are also low and so, do not affect overall energy. However, threshold voltage variation caused by Random Dopant Fluctuation (RDF) affects energy efficiency and circuit performance [14].

Since switching is done using leakage current in subthreshold, circuit performance is susceptible to threshold voltage variation and the leakage current variation. In addition, output signal integrity depends on the ratio of on-current (I_{on}) to off-current

(I_{off}), which is affected by threshold voltage variation. Equations 2.2 and 2.6 show the dependence of drain current and circuit delay on threshold voltage. Variation in leakage current caused by threshold voltage variation, impacts standby power while variation in delay impacts circuit performance.

Transistor sizing with energy constraints was used to mitigate variation [27]. It was shown upsizing at the expense of raising minimum energy and optimal supply voltage led to improved circuit robustness. Since random dopant fluctuation affects more or less all gates in a circuit path, having more gates in a path could have an averaging effect on the variation. Architectural approaches to mitigate variation in the form of deeper pipelines offering greater logic depth was evaluated [14]. Accessing SRAM bit cell using transmission gates instead of pass transistors was employed [15]. This single-ended cell in conjunction with write assist offered good noise margins in the presence of parametric variation.

2.2.2 Compensation using circuit techniques

Mitigating variability can also be addressed using circuit techniques which leverage process technology features. One such technique is body-biasing, which leverages the body-effect of an MOS-transistor. By applying a voltage to the body terminal, the threshold voltage of the transistor can be changed. This in turn changes the switching speed and leakage current of the transistor. The change in threshold voltage of an NMOS transistor, for an applied bias voltage is given by 2.17 [54],

$$\Delta V_{th} = V_{th0} + \gamma \cdot \left(\sqrt{|2\phi_F - V_{bs}|} - \sqrt{|2\phi_F|} \right) \tag{2.17}$$

where V_{sb} is the voltage at the body terminal with respect to the source terminal, V_{th0} is the threshold voltage with no body bias, V_{th} is the threshold voltage in the presence of applied body bias, ϕ_F is the Fermi potential of the bulk terminal and γ is the body-effect coefficient. By increasing the voltage at the body terminal of an NMOS transistor with reference to its source, the threshold voltage can be reduced, thus making the transistor switch faster. Conversely, by reducing the body voltage, the transistor switching is slowed and making it less leaky. Together, these techniques can improve circuit performance by selectively speeding up gates on the critical path and save power in unused functional blocks in circuits by making them less leaky.

However, there are limits to the benefits offered by body biasing. Reverse bias is less effective to reduce the subthreshold leakage of short-channel devices compared to unbiased devices [4]. Reverse bias was found to increase the band-to-band tunnelling leakage at source-drain junctions when the applied bias exceeds an optimum value [17]. Forward bias, on the other hand, reduced die-to-die parametric variations even in the presence of bias potential fluctuation. A combination of forward and reverse biases can be effectively used to mitigate process variation while meeting frequency and leakage requirements [29].

Body biasing as a compensation technique has been deployed often in circuits operating at peak supply voltage. Its benefits at subthreshold operation have not been widely explored. The impact of adaptive voltage scaling in mitigating variation was compared with adaptive body biasing [52]. It was found that adaptive body biasing was energy optimal compared to voltage scaling.

While these circuit applications and mitigation techniques helped establish subthreshold as a viable ultra-low power solution, they belong to either custom design methodologies or modified standard-cell design methodologies. These methodologies require extended design cycles involving mask-making and associated Non-Recurring Engineering (NRE) costs. There is a pressing need to investigate subthreshold circuit operation in methodologies beyond custom integrated circuits.

2.3 FPGA Methodology

Integrated circuit (IC) design continues to be dominated by full-custom design and semi-custom design including Application Specific Integrated Circuits (ASICs). So, IC design performance developed at a pace dictated by silicon foundries and their manufacturing processes. With the introduction of Programmable Logic [62] in the form of Complex Programmable Logic Devices (CPLDs) and Field Programmable Logic Arrays (FPGAs), this dependency was gradually removed, thereby enabling designers to accurately and quickly prototype their designs. By delinking the design phase from the manufacturing process, it was possible to speed up the design cycle and reach wider application domains. Programmable Logic gained popularity because the end-user could configure or program the circuit in the field to achieve desired circuit functionality, unlike custom designed silicon or ASICs. Moreover, programmable logic did not have significant Non-Recurring Engineering (NRE) costs like lithographic mask-making associated with custom designed silicon. Some of the functionality accomplished by a

sub-set of such masks was instead transferred to end-user in the form of configurability. As result, the earliest adopters of this technology included domains with requirements of in-field programmability, namely, automobiles, satellite technology, instrumentation and industrial automation.

2.3.1 Design Flow

In an FPGA methodology, a pre-manufactured silicon fabric is made available, which is programmed to perform a desired logic functionality. The uncommitted silicon fabric typically consists of an array of logic blocks and interconnect blocks. By the process of programming, also known as configuring the fabric, the circuit is ready for operation and performs logic functionality as any other custom designed silicon. The important feature of programmable devices is that, a different circuit functionality can be implemented on the same silicon fabric, simply by reprogramming the device. This offers a great deal of flexibility. This is one of the major advantages in this methodology.

In an FPGA methodology, the design process starts with design entry using a schematic-based editor or using a Hardware Description Language (HDL) based description of the logic design information. Using manufacturer provided proprietary Computer Aided Design (CAD) tools, a netlist description of the circuit logic functionality is created from the initial description. This description usually takes into account the number of available logic and interconnect blocks, collectively known as the resources of the FPGA fabric. At this stage, the netlist description is an optimized version of the user described logic. The netlist is then physically implemented on the FPGA fabric

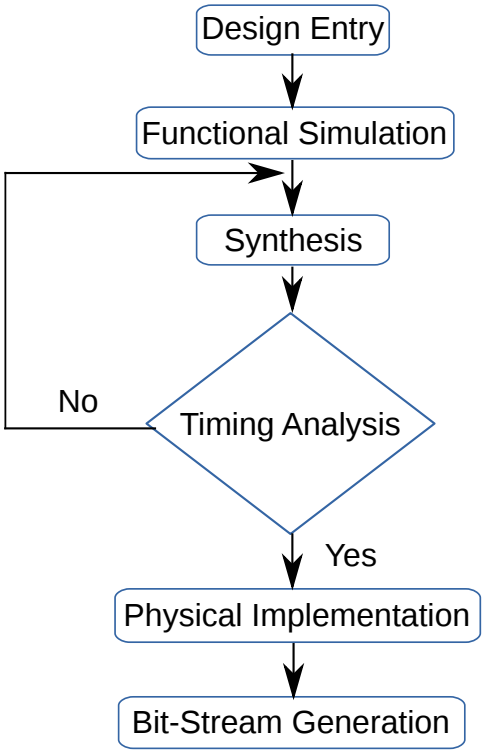

Figure 2.2: Figure showing typical FPGA Design Flow

using proprietary or open-source CAD tools and estimates of physical implementation are conveyed to the user. At this stage the user can iteratively meet target functionality and performance.

The actual process of configuring the fabric for the design application is done by selectively programming the logic and interconnect resources. This programming involves writing onto memory cells to realize select functionality. The scope and functionality of the CAD tools at each stage of this design flow varies based on the FPGA vendor. Figure 2.2 shows a typical FPGA design flow as discussed.

FPGAs while being very effective for rapid prototyping have some disadvan-

tages. The uncommitted logic and interconnect resources on the fabric cannot be eliminated and thereby consume standby power. Previous research on reducing power consumption in FPGAs has focused on: select line control to unused multiplexers [59], body-bias to adaptively compensate process variation and reduce leakage [43], input vector reordering [24], body-bias together with multi-Vt technology [5].

Although FPGAs offer certain advantages, there are some trade-offs that need be to considered, when adopting an FPGA methodology. First, the FPGA fabric is manufactured for a class of applications and so, any specific circuit application uses a subset of available resources on the chip. The unused silicon is an area overhead, translating to die-cost. Moreover, unused circuitry consumes standby power, which must be factored in by the designer. Second, although FPGAs enable rapid prototyping, the power and performance achieved by an equivalent ASIC/Custom-silicon implementation typically will be significantly better. Third, for FPGA applications containing only volatile memory sub-systems, the need for reconfiguration whenever the power supply is switched off, can quickly become an overhead.

In design applications that use only a small portion of available resources the standby power can become a significant component of total power. This has been a barrier for FPGAs in the ultra-low power design space. For low-volume applications which cannot afford to produce custom silicon, yet require the flexibility of programming and are energy constrained, an FPGA capable of operating in subthreshold voltages can be the solution.

2.3.2 FPGAs in Subthreshold

FPGAs capable of operating in subthreshold voltages combine the benefits of rapid prototyping with energy savings orders of magnitude greater than FPGAs operating at normal supply voltages.

The building blocks of FPGAs typically include configurable logic blocks and configurable interconnect. These blocks employ circuit topologies like pass-transistors, flip-flops, multiplexers and Static Random Access Memory (SRAM). Operating these circuits at subthreshold requires design techniques different from that of normal supply voltages.

Previous research on subthreshold FPGAs has leveraged the use of clustered logic block elements [51, 21]. Architecturally, clustered logic blocks enable reduced consumption of interconnect resources. Typically, in FPGAs, the circuit paths of logic blocks is longer than that of the routing resources. With clustered logic blocks, the delay through such paths is increased and as a result the active energy consumption could be higher. A subthreshold FPGA using a 6-transistor interruptible-latch instead of conventional 6-transistor SRAM cells was designed [21]. However, in this latch, the cross coupled feedback is delivered back through a pass transistor topology. The threshold voltage drop and sneak current path typically occurring in pass transistor topologies could compromise the integrity of the stored data in the latch. A subthreshold FPGA with two separate supply voltages, one for logic and another for the configuration memory was designed [51]. Having additional supply rails leads to implementation

complexity. Although high clustering reduces the interconnect latency, the logic path becomes longer.

While these implementations brought focus to subthreshold FPGAs, they did not have compensation mechanisms to mitigate impact of process variation on circuit performance.

Chapter 3

Subthreshold FPGA Design and Implementation

One of the contributions of this thesis is the design of a subthreshold FPGA core with body bias voltage assignment to compensate against process variation. Design of a new FPGA core is a custom design process integrating architectural specification, block and sub-system design, design automation and configuration bit-stream. This chapter presents the architecture, design and implementation of a single V_{dd} subthreshold FPGA chip. The design automation methodology required to implement and verify the design, and Printed Circuit Board (PCB) design to test the chip are also discussed.

3.1 Architecture

Over the years several FPGA architectures have been developed. The notable ones include include row-based architecture [9], sea-of-gates architecture [10], hierarchical

architecture [66] and island style architecture [63]. Of these, this work adopts the island style architecture considering the availability of supporting sotware tools in the public domain.

3.1.1 Island Style Architecture

Architecturally an FPGA consists of an array of configurable logic interspersed with routing resources and one or more high-speed transceivers. In the island-style architecture the logic blocks are surrounded by switch blocks and connection blocks, which are together referred to as routing resources. Although most modern FPGAs include transceiver circuit modules, to enable high speed data transfer to and from the FPGA, the scope of this research is limited to the core programmable fabric. Figure 3.1 shows a conceptual figure of a FPGA conforming to the Island-Style Architecture.

3.1.2 Architectural Exploration

In developing a new FPGA fabric, architectural considerations that can impact power and performance need to evaluated early in the design phase. This typically includes the capacity of logic blocks and routing resources. While the properties of the individual blocks can be estimated early in the design phase of the new fabric, their interconnected performance can be estimated only by the use of tools designed specifically for such architectural exploration. This work uses FlowMap [26], T-VPack and VPR [16], which collectively enable FPGA architecture exploration. Using VPR, the performance of architectures with novel logic block and switching topologies can be es-

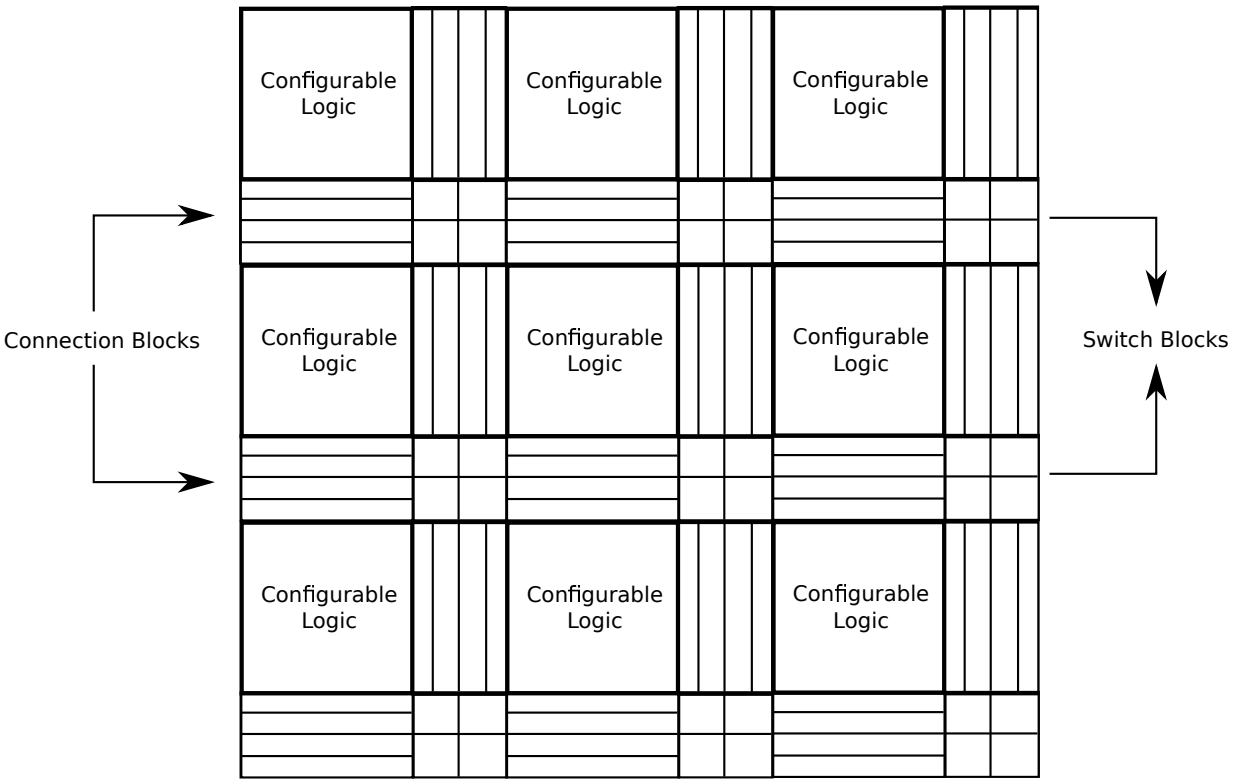

Figure 3.1: Island-Style architecture containing an array of configurable logic blocks surrounded by interconnect

timated. Functionally, there are two inputs to this methodology. First, a description of the proposed FPGA architecture specifying the capacities of the logic blocks, switching boxes and routing channels. The other input being, a netlist description of standardized benchmark circuit applications. The output from VPR is a description of the placed and routed netlist of that circuit application on the proposed FPGA architecture. The resulting routing description needs to be implemented using actual transistor models for functional simulation. Simulation of such a transistor level model of the routing description yields the functionality of the original circuit application.

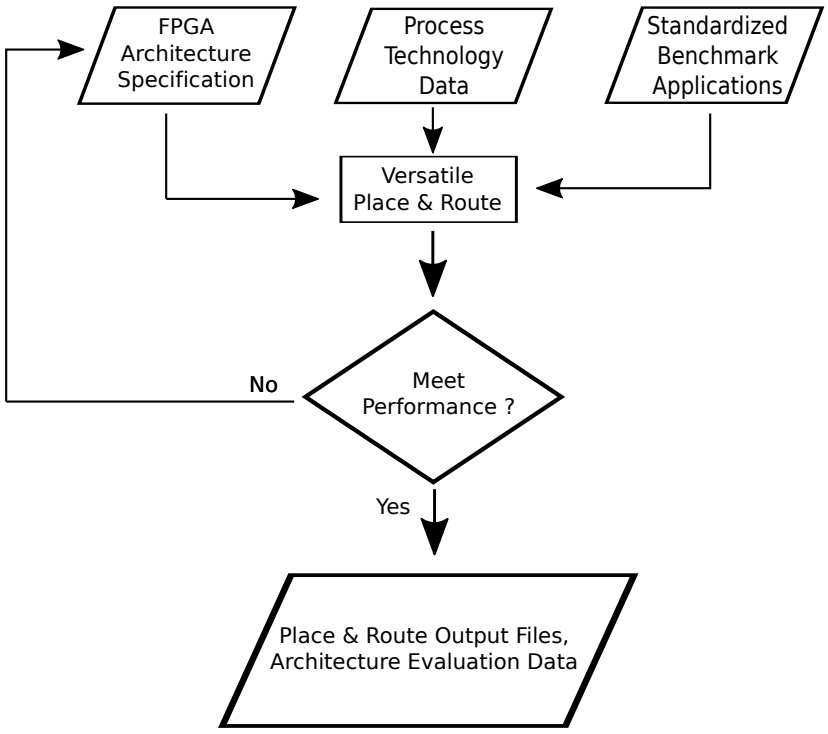

Figure 3.2: Architectural exploration of new FPGA architecture using Versatile Place and Route

3.1.3 Specifications

3.1.4 Programmable Logic

The granularity and size of the programmable logic and interconnect play an important role in the performance and area of the FPGA fabric. Programmable logic typically consists of Look-Up Tables (LUT) which can be used to implement Boolean functions. These logic blocks can also be clustered together to map complex Boolean functions. Clustering of logic blocks can lead to savings in routing area [28]. Logic blocks containing 4-input LUTs were found to be optimal for area and for performance. To tape-out

this subthreshold FPGA, silicon die-area was a key constraint, leading to design decision of clustered logic block architecture. Each cluster contained 4 logic slices, each implemented using a 4-input LUT. Figure 3.3 shows a representative 4-input Look Up Table based logic block.

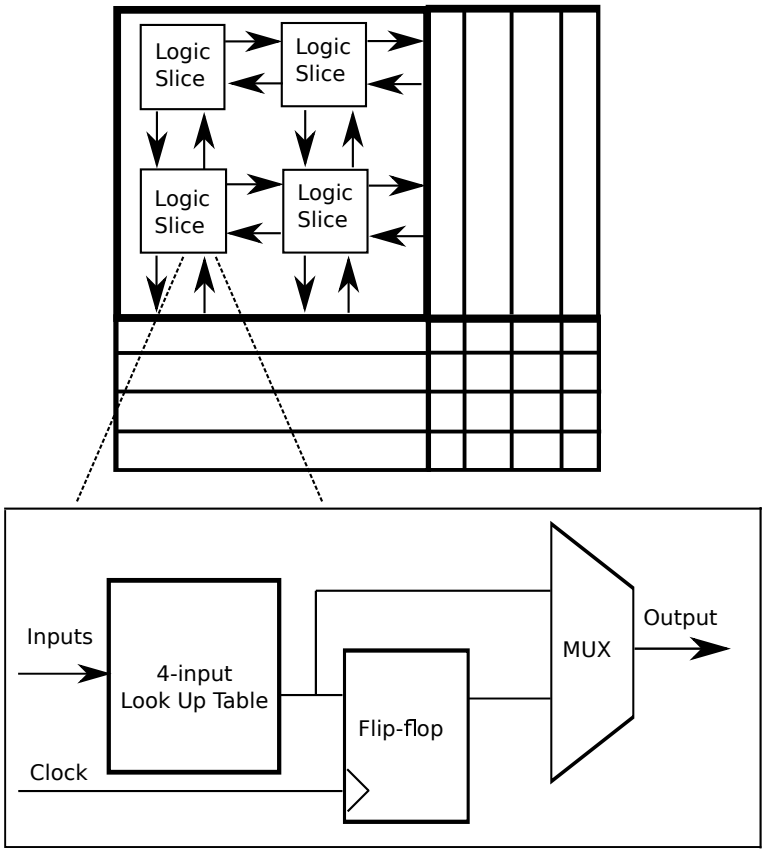

Figure 3.3: A representative logic block containing 4-input Look Up Table constitutes a logic slice

3.1.5 Programmable Interconnect

Programmable Interconnect consists of routing tracks or channels and switch blocks, both of which are programmable. The channels run in both axes along the length and

breadth of the fabric. At the intersection of any two channel segments, a switch block serves to perform directional routing. The sizes of channel segments and that of the switch blocks must match. The ability to route any circuit application on an FPGA fabric is collectively determined by the available tracks in a channel and the flexibility of the switch blocks. Channel segments with 12-tracks and switch blocks with flexibility of 3 were found to provide sufficient routability. We chose channels with 12 tracks and switch blocks with a flexibility of 3 for our programmable interconnect. Figure 3.4 shows a representative connection block and switch block respectively

3.2 Block Designs

3.2.1 Process Technology

This chip was designed using an IBM CMRF8SF [36] 120 nanometer process technology and fabricated at Metal Oxide Semiconductor Implementation Service (MOSIS) using the MOSIS Educational Program (MEP) [38] initiative. Some of the key features available in this process node include bulk CMOS with triple-well FETs, 8 metal layers, core transistors operating at 1.2V and I/O transistors operating at 3.3V. The nominal threshold voltage of the NFETs and PFETs are respectively 0.350V and -0.350V.

3.2.2 Body biasing

The biggest challenge in designing circuits for subthreshold operation is the impact of process variation. To mitigate and compensate against process variation, this de-

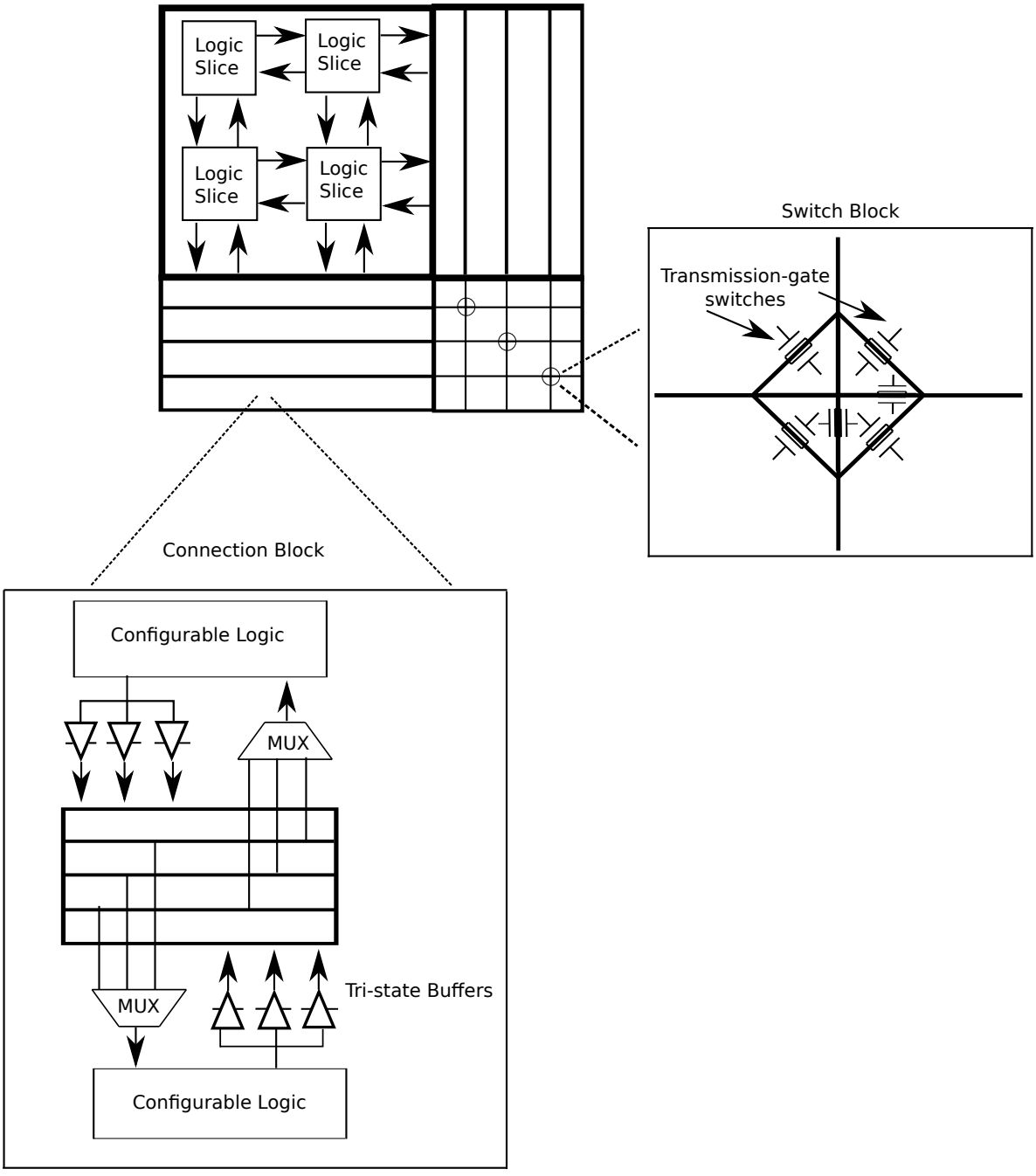

Figure 3.4: A representative switch block and 2 instances of connection blocks constitute a programmable tile

sign methodology uses body-bias technique. This is implemented by designing circuit topologies using triple-wells and hierarchically connecting them together to form a bias net.

3.2.3 Cell Library Design

The programmable logic and interconnect in this FPGA consists of a set of basic circuit topoplogies reused across different functional blocks. To implement this FPGA architecture a cell library was developed consisting of multiplexers, transmission gates, tri-state buffers, inverters and flip-flops designed to work at subthreshold voltages.

Multiplexers, tri-state buffers and inverters were designed for rail-to-rail operation. Transmission gates were used instead of pass transistor gates, which cannot function at subthreshold voltages. Master-slave latches containing transmission gates were used for flip-flops to enable subthreshold operation. Each cell of this library was verified by extensive transistor level simulation at subthreshold voltages.

3.2.4 Logic Block

The programmable logic consists of configurable logic blocks, made up of a cluster of logic slices. Each of these logic slices is identical in function and capacity. The logic slices are implemented using a Look Up Table (LUT), flip-flop and multiplexer. Using a 4-input LUT any Boolean function with upto 4 variables can be mapped. The flip-flop is optional and can be configured to register the output of LUT. The multiplexer can be configured to select between the output of LUT or flip-flop. Figure 3.3 shows a

representative logic block built using 4 logic slices, with each slice implemented using a 4-input Look Up Table.

3.2.5 Connection Block

The connection block serves to interconnect any logic block with other logic blocks and with the primary inputs and outputs of the circuit. The connection blocks consist of tri-state buffers and multiplexers. The tri-state buffers can be configured to connect the outputs of logic blocks on select routing tracks. The multiplexers can be configured to connect select routing track to the inputs of logic blocks. Figure 3.4 shows a connection block which can connect the output of logic blocks to routing tracks and routing tracks to logic block inputs.

3.2.6 Switch Block

The switch block consists of a matrix of programmable switches. Switch blocks serve to connect connection blocks and to provide directional routing. Each track in a channel requires a programmable switch to connect onto a track in another channel. The programmable switch consists of a group of 6 transmission gates wired together to provide a switching flexibility of 3. A switch flexibility of 3 means a track entering a switch block from the north direction can be configured to connect to a track in west, south or east directions. Figure 3.4 shows a switch block with 3 representative switches, each with a switching flexibility of 3 directions.

3.2.7 Programmable Tile

As part of a hierarchical design process, a programmable tile consisting of logic and interconnect blocks was designed to enable implementation of the full FPGA fabric through simple replication of the tile. This tile consists of one clustered logic block abutting a connection block in the vertical orientation and a connection block in the horizontal orientation with a switch block connected between these connection blocks. This design can connect the inputs and outputs of the logic block to the routing. For an array of programmable tiles, the routing interconnects logic blocks to other logic blocks or to primary inputs and outputs. Figures 3.3 and 3.4 show a programmable tile and the constituents of the tile, namely logic block, connection block and switch block.

3.2.8 Configuration SRAM

Circuit functionality in programmable logic and interconnect is enabled by configuring the content of Static Random Access Memory (SRAM) cells. At subthreshold voltages, six-transistor (6T) SRAM cells have issues with read-stability and write-stability. To attain SRAM functionality, a modified architecture is used in this work. Subthreshold SRAM operation is limited by stability factor, particularly during the read operation. The configuration SRAMs required for the programmable tile only need be written and not accessed for read. Unlike typical SRAM array operation involving WRITE and READ cycles, the bits needed by the tile only have to be written. The stored content continually is used by the programmable circuitry without having explicit READ oper-

ations. This work uses the write-assist technique [42] but customizes the architecture for the array-like structure in this FPGA. A write-assist PFET is shared by the SRAM cells in each row of a tile.

In this tile-based architecture, configuration cells are assigned to logic and interconnect blocks to program the SRAMs on a tile-basis. The logic blocks consist of 4-input LUTs, which required 16 SRAM cells to realize upto 4-variable functionality. The clustered logic blocks consisting of 4 logic slices required 64 SRAM cells. Each connection block required 10 SRAM cells. Each programmable switch required 6 SRAM cells and a programmable switch block required 73 SRAM cells. The total number of SRAM cells per tile was 192.

Figure 3.5 shows a reduced representation of the configuration cells in a programmable tile.

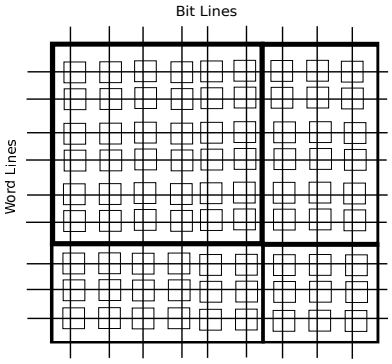

Figure 3.5: Configuration SRAMs of a programmable tile

3.2.9 Scan Block

Scan-design is a test methodology intended to provide controllability and observability of the registers in a circuit. By accessing the registers using this method, the combinational logic situated away from chip inputs and outputs can be observed.

A scan-chain consists of instances of a specially designed flip-flop connected together in series. Figure 3.6 shows a conceptual illustration of a scan flip-flop. Each

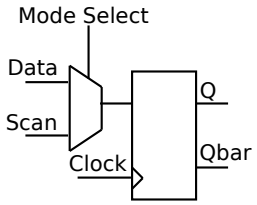

Figure 3.6: A Scan flip-flop can operate in two modes.

flip-flop has a multiplexer driving the D-input of the flip-flop. The select line of the multiplexer can be used to choose between scan-mode and normal-mode of operation for the flip-flop. In the normal mode, the scan flip-flop functions as a regular D flip-flop. When all flip-flops in a scan-chain are in the scan mode, a desired bit-pattern can be loaded in by clocking a specific number of cycles. Subsequently, when the scan-chain is reverted to normal mode, the clocked in bit pattern is available to perform computation.

The ability of scan-chain to make data available at registers is leveraged to input configuration data in this FPGA architecture. The scan-chain design for this FPGA consisted of a scan-block whose outputs drive the word-select line of the programmable tile and a scan-block whose outputs drive the bit-lines and bit-bar-lines of

the programmable tile. Since the array of tiles are connected by abutted placement, the word-lines of all tiles in a given row area also connected together. Each programmable tile on the periphery of the chip abuts a scan block, whose output drives the wordline of a given row. A similar technique is used for the bit-lines with its dedicated scan blocks.

Using these word-line and bit-line scan chains, bit sequences to configure the content of SRAMs were clocked in. Figure 3.7 shows a conceptual illustration of a programmable tile on the chip periphery supported by scan chains for word-lines and bit-lines.

3.3 Design Automation

The design automation developed as part of this work consisted of a design implementation flow and a bit-configuration flow. Using these two flows, the following goals were collectively achieved:

- Use the cell level designs to implement FPGA core fabric.
- Verify design intent against design implementation.
- Develop foundry collateral for silicon tape-out using design implementation.
- Develop bit-configuration to enable benchmark circuit synthesis on proposed FPGA.

The purpose of the design implementation flow was to implement the largest possible FPGA fabric design given die-area constraints by leveraging the cell library. Since the FPGA consists of an array of programmable tiles, the optimal approach was to design

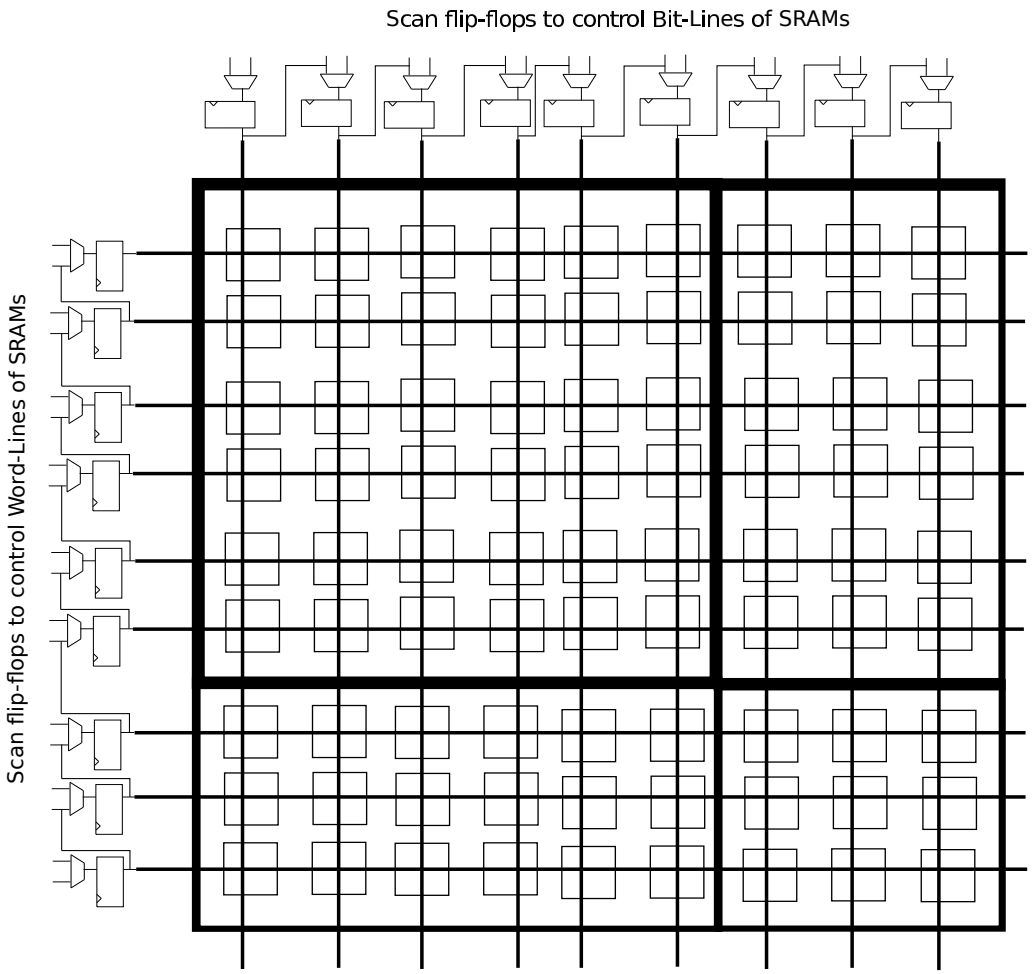

Figure 3.7: Each programmable tile on the top and left sides of the chip periphery is supported by a scan block of the scan chain.

a single tile and replicate it, to obtain the complete FPGA core. By performing this at two levels of design abstraction, namely netlist-level and layout-level, the first three goals were achieved. Figure 3.8 shows the design implementation flow.

The Design Implementation flow uses the Cadence Design Framework II (DFII) environment to design and implement the cell level designs, using the CMRF8SF 120 nanometer process technology. Using the cell library, the configurable logic block, con-

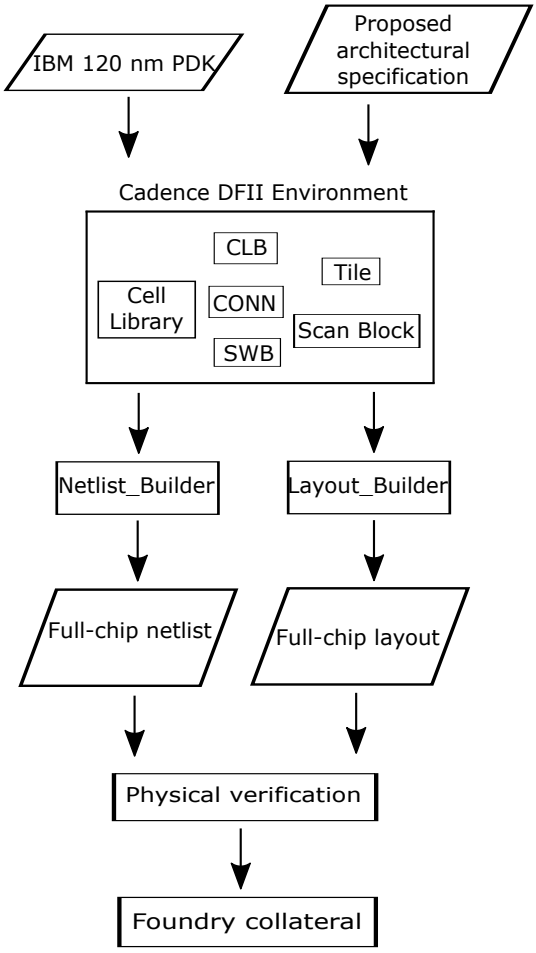

Figure 3.8: Design Implementation flow

nection block, switch block, scan block and programmable tile were designed in the DFII environment. 'Netlist_Builder' was a utility developed in Python language to build and connect the array of logic and interconnect blocks along with the required scan blocks, at the netlist level of description. 'Layout_Builder' was a utility developed in Python to build the layout level description of the FPGA. This utility was a wrapper around a script developed in the SKILL language available as part of the Cadence DFII environ-

ment. This full chip layout was then checked for Design Rule Check (DRC) violations, to prepare it for foundry delivery. Since the netlist description of the FPGA and its laytout description were obtained using two different utilities, it is required to verify the integrity of the implementation. This was accomplished by performing a Layout Versus Schematic (LVS) check. Upon passing the DRC and LVS checks, the full chip layout was delivered to MOSIS for fabrication.

To enable synthesis of circuit applications on proposed FPGA fabric and to configure the chip, a bit stream is required. This was accomplished by developing a Bit-Stream Configuration Flow. Figure 3.9 summarizes the Bit-Stream Configuration Flow. This flow uses as inputs, the placed and routed description of a given benchmark from

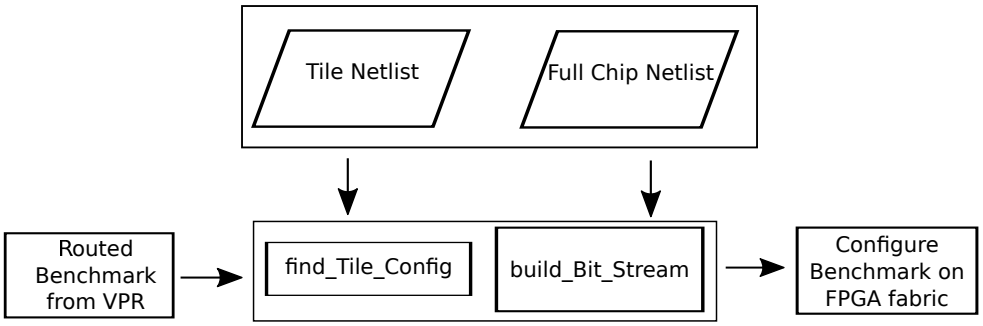

Figure 3.9: Bit Configuration flow

VPR, tile sub-circuit description at the netlist-level and full chip circuit description at the netlist-level. The utility 'find_Tile_Config' identifies the configurations required for each tile of the fabric to achieve the functionality described in the routing description. The utility 'build_Bit_Stream' develops the bit stream taking into account the physical locations of the different tiles by interleaving the individual tile configurations.

3.4 Printed Circuit Board Design

A Printed Circuit Board (PCB) was designed, to mount the fabricated FPGA chip and configure it for testing purposes. This board design was based on open-source hardware Arduino Leonardo and customized to work with the subthreshold FPGA. The board consists of an 8-bit microcontroller which can be programmed using the Arduino development environment and configured through a USB interface. This microcontroller in turn drives the programming pins of the subthreshold FPGA. The overall goal is to program the microcontroller to output the bit-stream required to configure the FPGA for a desired benchmark.

Figure 3.10 shows key details of the PCB design. It should be noted the actual PCB design includes typical components like capacitors and jumpers to group power supply pins and signal traces required for board implementation purposes and are not discussed here.

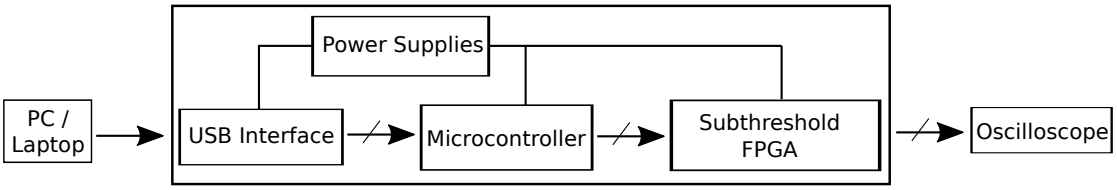

Figure 3.10: PCB High Level Schematic

Table 3.1: Summary of benchmark place and route on proposed FPGA fabric

Circuit	Placed	Routed
c17	✓	✓
c432	✓	✓
c1908	✓	×
s344	✓	✓
s349	✓	✓
s382	✓	✓
s400	✓	✓
s444	✓	✓
s526	✓	✓
s953	✓	×
s1196	✓	×
s1423	✓	×

3.5 Results

This section discusses the outcome of synthesizing select benchmark circuits on the proposed FPGA architecture and chip testing.

3.5.1 Benchmark Synthesis

Benchmarks from the ISCAS85 collection were used to perform synthesis, place and route on this FPGA architecture. The FPGA fabric contained 198 programmable tiles as an array of 11 rows and 18 columns. Table 3.1 shows the summary of synthesis, place and route on select benchmarks. In the cases where routing did not succeed, it was due to the limit of available routing resources.

3.5.2 Chip Details

The full chip layout as sent to foundry for fabrication is shown in Figure 3.11. For visual clarity, only the top few metal layers are displayed. The dimensions of the chip are 4

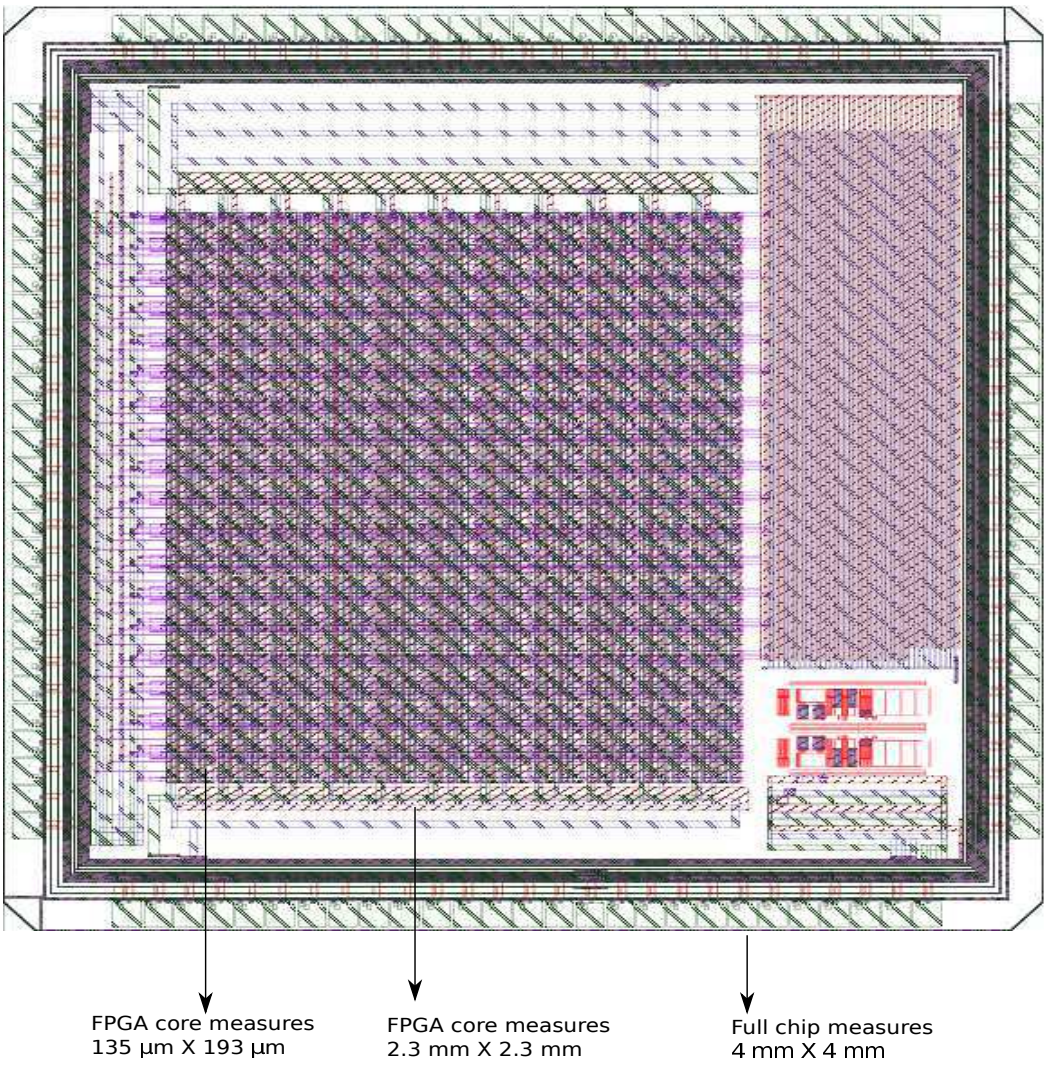

Figure 3.11: Full chip layout

mm x 4 mm. The dimensions of the programmable logic core are 2.3 mm x 2.5 mm.

Each programmable tile measures 135 um x 193 um. The FPGA fabric consists of an array of 11 rows x 18 columns of programmable tiles.

This chip was packaged in a Kyocera 108 pin Open Ceramic Pin Grid Array (PGA) package [37].

3.5.3 Area overhead

Triple well process and accessing the wells incurs an overhead. The routing overhead for additional rails to bias the N-wells and P-wells of all the logic blocks and routing structures is 6% of the overall chip area. To measure area overhead when using triple wells and body bias methodology, an area estimate of an 8-input multiplexer is adopted. Since this cell from the cell library is used across logic and routing blocks, this provides a high level estimate. Using this 8-input multiplexer it was found triple well and body bias methodology incurs approximately a 24% area overhead, compared to non-triple well processes.

3.5.4 Chip Testing

This subthreshold FPGA chip was tested using the PCB-setup containing the microcontroller. It was found the FPGA had functional issues. So, the power and performance of the benchmark circuits could not be measured on silicon. Figure below shows the PCB assembly.

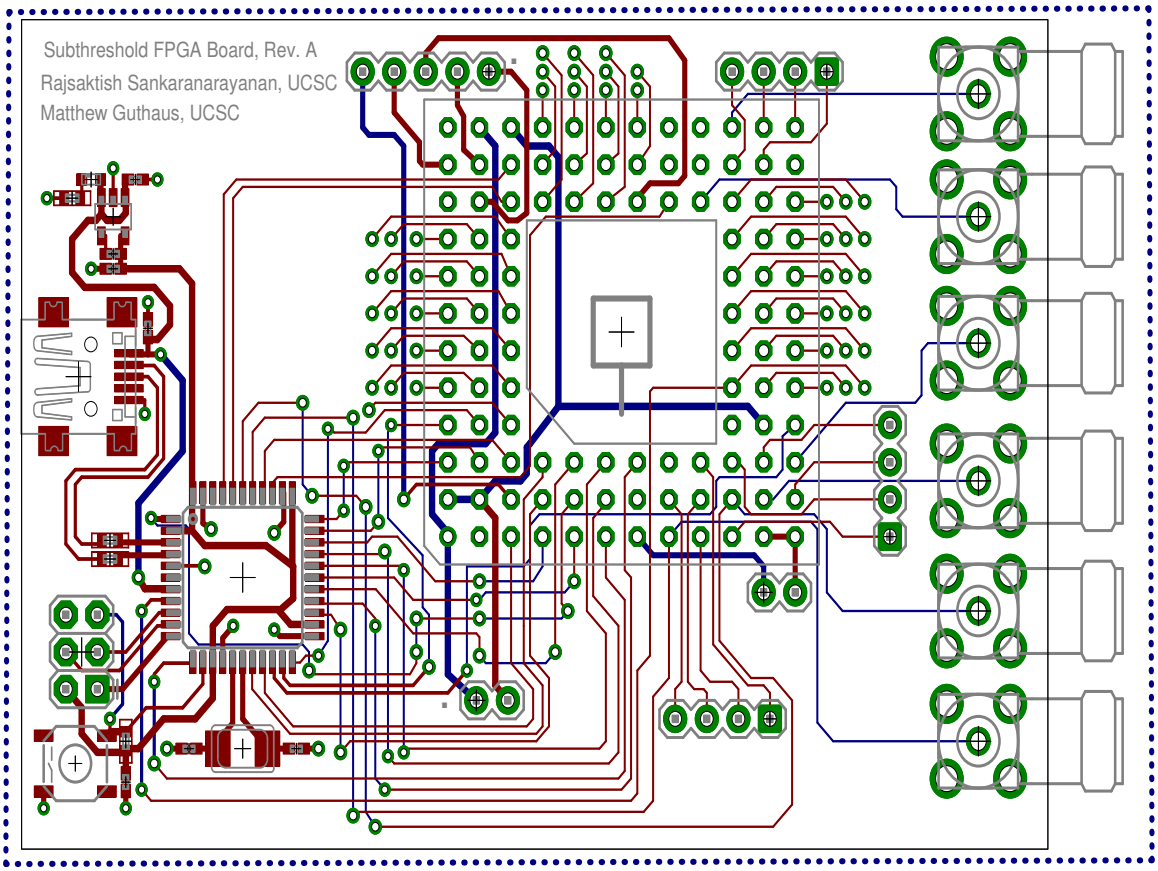

Figure 3.12: PCB layout

Chapter 4

Energy Optimality in Subthreshold FPGAs

To develop reliable subthreshold FPGAs as a solution for highly energy constrained applications, determination of energy savings by trading-off performance and sensitivity to process variation are required. The second contribution of this thesis is the determination of energy-performance trade-off, energy optimality and block-level energy consumption in FPGAs. This is accomplished using a simulation and characterization methodology. This chapter discusses the FPGA architecture used in this approach, the characterization framework and the energy analysis methodology. This chapter also analyzes the impact of variability on energy and performance, and evaluates the effectiveness of body bias in mitigating the sensitivity to variation.

4.1 Architecture

This methodology uses an FPGA conforming to the island style architecture discussed in Chapter 3.1.1. The availability of public domain software tools is a major factor in adopting this architecture. The block level designs are broadly based on the specifications discussed in Chapter 3.1.3. However, considering the goal of this methodology, that is energy optimality, certain design changes have been incorporated.

4.1.1 Design Implementation

The programmable logic and interconnect in this FPGA were implemented using a hierarchical design methodology starting with leaf cells. The technology used for implementing this design was a Taiwan Semiconductor Manufacturing Company (TSMC) 180 nanometer process supported by MOSIS [39]. For this process, the transistor threshold voltages are 0.360 V and -0.400 V for NMOS and PMOS, respectively.

Logic Block

The programmable logic block consist of a 4-input Look-Up Table (LUT), a multiplexer and an optional flip-flop similar to the logic block discussed earlier in Chapter 3.1.4. To gain insight about the power and performance of logic block and its impact on the overall power, a non-clustered logic block was chosen.

Connection Block

The connection block design closely resembles the design discussed earlier in Chapter 3.2.4. For this work, the connection blocks are 6 tracks wide.

Switch Block

The programmable switches are identical to to the design discussed earlier in Chapter 3.2.6. Since the connection block track width and the switch block size must match, the number of programmable switches is also 6.

4.2 Circuit Characterization

To implement the functional blocks, a cell library was developed by custom-designing and laying out the cells. This included logic gates, multiplexers, transmission gates and flip-flops. The netlists containing parasitic information were extracted for each block and verified by simulation. Each block was connected to appropriate other blocks to represent the loading capacitances when placed in an array like structure in a FPGA fabric. Power and performance measurements are obtained by simulation using Synopsys Hspice [60]. The functionality of each block is configured with a bit-stream and then simulated for a fixed number of cycles. The configuration bit cells operate at subthreshold voltages.

It is assumed that the configuration phase is infrequent compared to the operating phase and therefore we disregard analysis of this.

The logic simulation input uses random data with a range of toggle activity factors with respect to the clock toggling rate. Although low power applications typically have low activity factors [61], we chose a range of 25-100% to comprehensively determine the circuit response to occasional higher demands of input activity.

At each supply voltage, the blocks are analyzed at the maximum clock frequency with which the block can reliably function. The supply voltages range from subthreshold ($\approx 110mV$) to super-threshold ($1.8V$). The constituent blocks were individually characterized for performance and power. By characterization it was found, the blocks were functional at voltages as low as $110mV$. So, we chose this as the lower range for our analysis.

4.2.1 Benchmark Circuit

This analysis uses a representative benchmark c432 from the ISCAS-85 set of benchmarks. This analysis was intended as a proof-of-concept showing energy-performance trade-off, by performing high-level energy estimation in an island style architecture. Moreover, from a thesis timeline perspective, this analysis preceded the actual chip design and tape-out. Starting with a .blif description of the circuit, the placed and routed net-list was obtained using a tool flow involving FlowMap [26], T-VPack and VPR [16]. The FPGA fabric chosen for this experiment was an array of 11 X 11 containing 121 logic blocks and supporting routing blocks. This experiment preceded the chip tape-out discussed in Chapter 3. and this size (11 X 11) was the initial estimate for chip tape-out. The block level timing information given to VPR, for each supply voltage of

our analysis comes from the corresponding characterization data for that supply voltage. This resulted in distinctly different critical path delays across the range of supply voltages. However, routing track width and number of logic resources available to VPR was fixed (11 X 11), thus representative of silicon. This ensured fairness of benchmark comparison across supply voltages. It must be noted that in each case the benchmarks were routable. Across the range of supply voltages, the utilization of logic and routing resources was above 90%.

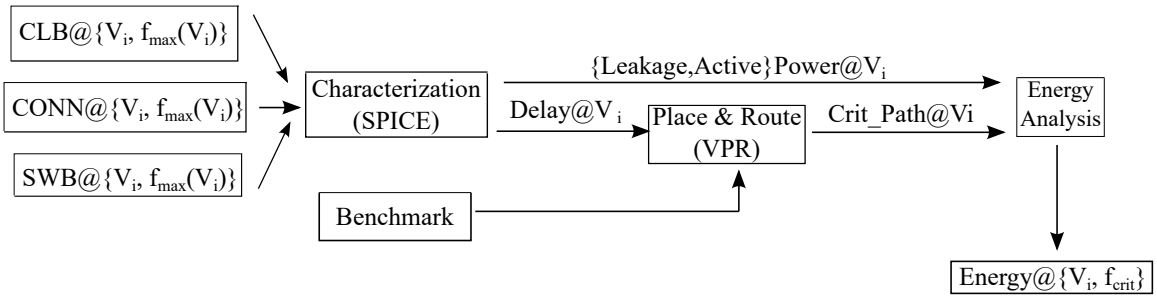

Figure 4.1: Proposed methodology to perform block-level Characterization benchmark P&R and Energy Analysis.

4.3 Energy Analysis Methodology

Using the characterization data obtained from block level Hspice simulations and the critical path delay of the placed and routed benchmark at each supply voltage we computed the energy of the benchmark.

Let X, Y, and Z be the number of Logic block, Connection block, and Switch blocks, respectively, in the benchmark. At a supply voltage, the power delay product

of the fabric is given by

$$(X \cdot Pwr_{CLB} + Y \cdot Pwr_{CONN} + Z \cdot Pwr_{SWB}) \cdot Delay_{CritPath} \tag{4.1}$$

Now let us consider the constituent blocks individually, for the sake of clarity. In 4.1, the power consumed by CLB assumes the frequency at which it was characterized. This may be different from the frequency at which the benchmark runs. So, we correct for this using

$$(X \cdot \frac{Pwr_{CLB}}{freq_{CLB}} \cdot freq_{CritPath}) \tag{4.2}$$

for each block where $freq_{CritPath} = \frac{1}{Delay_{CritPath}}$. Applying to all blocks in 4.1, the shared critical path frequency cancels out with the critical path delay obtaining

$$(X \cdot \frac{Pwr_{CLB}}{freq_{CLB}}) + (Y \cdot \frac{Pwr_{CLB}}{freq_{CLB}}) + (Z \cdot \frac{Pwr_{CLB}}{freq_{CLB}}) \tag{4.3}$$

for the power delay product.

4.4 Reducing variation in energy using body-bias

Circuit switching in subthreshold is done using transistor leakage current. As a result, circuit performance and active energy are affected by leakage current, which depends exponentially on threshold voltage. To analyze the effect of parametric variation on FPGA performance and energy, the sensitivity of an 8-input multiplexer is measured. Since the 8-input multiplexer is a reused cell across connection blocks and logic blocks, this analysis provides a representative view of the FPGA. To consider the performance of a FPGA fabric in entirety, benchmarks need to be synthesized onto the fabric and suitable

test vectors developed. Instead, a cell which is reused across all blocks can provide an overview of the intended analysis. With this consideration, an 8-input multiplexer was chosen.

Typically body-biasing has been used as a post-silicon compensation technique to meet target power and performance. This is done by applying forward bias to speed up the slower dies and reverse bias to slow down the dies that consume excess power. Other approaches include applying specific bias compensation to P-wells and N-wells and maintaining an offset between them [52]. As a design strategy, body-bias application involves determination of optimal bias required for a given die or region of a given die. This is accomplished using dedicated analog or digital feedback circuitry controlling bias generators [29] Other strategies include Low Voltage Swapped Bias (LVSB) which tie all N-wells to ground and all P-wells to supply voltage [58]. The efectiveness of body bias is determined by: bias circuitry energy overhead, bias circuitry area overhead, scalability of bias strategy to work in conjunction with other power saving techniques.

Since FPGAs are array-like structures, dedicated circuitry is required for each tile or a region composed of a number of tiles. However, this can lead to significant area overhead even when biasing one type of well [41]. Moreover, the bias voltages for N-wells and P-wells and the bias offset between the wells can have an infinite number of combinations to meet a target power or performance. A simpler approach is to bias both wells by the same quantity. Considering scalable operation at multiple subthreshold voltages, this work uses a forward body bias of $V_{dd}/2$.

The threshold voltages of NFETs and PFETs in the 8-input multiplexer are

assumed to be normally distributed around the nominal value with 1-sigma variance of 15mV [65]. All transistor instances are subject to threshold voltage variation from this distribution. Since circuit performance is determined by the critical path delay and subthreshold circuits are highly sensitive to threshold voltage variation, the critical path needs to be estimated for a given instance of applied variation. Circuit active and standby energy are measured by simulating the circuit using this operating frequency, obtained from the critical path delay. This process is repeated for 300 random Monte Carlo iterations.

4.5 Results

In this section we discuss the results from multiple standpoints including energy optimality, influence of switching, stand-by energy and delay analysis. Together, these provide a holistic picture about the trade-offs through subthreshold operation.

4.5.1 Overall Energy Optimality

Active power can be arbitrarily reduced by lowering the frequency of operation, switching activity in circuit, supply voltage or a combination of these. However, this may require additional operating time, and thus consume more energy to perform a computation. So, the energy to complete a fixed computation is a more relevant metric.

For each supply voltage, we measure the critical path delay of the circuit for that supply voltage. The inverse of the critical path delay gives the maximum operating frequency of the circuit. Operating the circuit at any frequency lower than the maximum

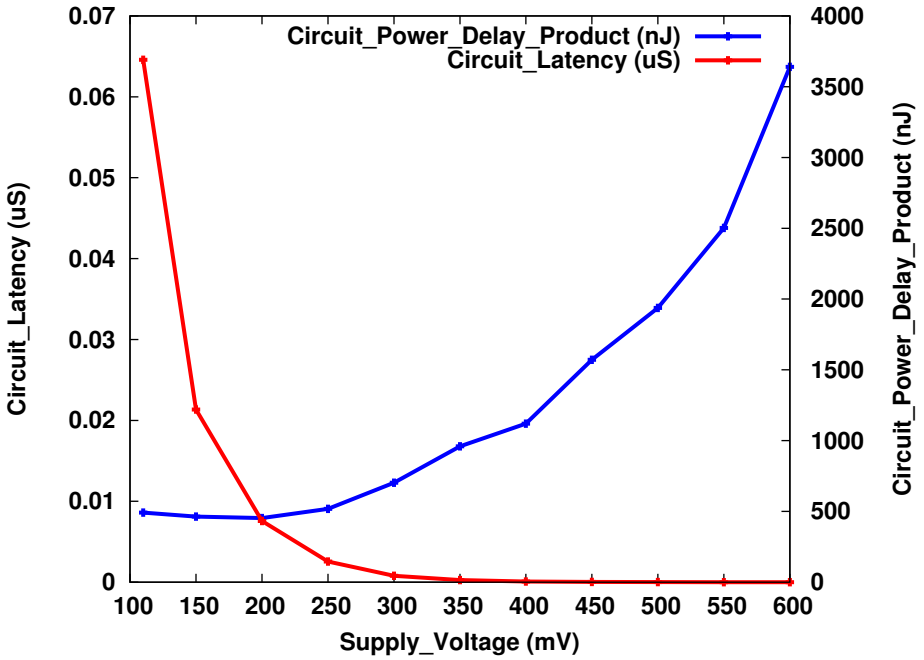

Figure 4.2: The minimum energy point and circuit latency trade-off over a range of supply voltages for benchmark c432 shows that the optimal energy point is around 200mV.

operating frequency is an under-utilization of circuit performance and consumes more energy than needed. We directly calculate the average power delay product (energy) per cycle from the measured power and period. Figure 4.2 shows the scaling of energy and circuit latency over a range of supply voltages. For purposes of clarity the X-axis shows only up to 600mV.

It can be seen from Figure 4.2 that there are two distinct regions of observation, namely, subthreshold region which is up to 360mV and super-threshold which is above 360mV. In the super-threshold region, the energy consumption grows exponentially, while the circuit performance has almost leveled off. This is due to the transistors crossing over from linear to saturation region and any raise in supply voltage causes

an increased power consumption without significantly improving the performance. This leads to the exponential energy growth. In the subthreshold region the drop in energy is much slower, almost quadratic and accompanied by a growing increase in circuit latency. At 200mV, the circuit has the minimum energy point, after which the energy consumption starts rising again, though slowly.

At supply voltages below the minimum energy point, the large delay of the transistors causes the circuit to perform computation over a long time, thus causing more energy consumption. At voltages higher than the minimum energy point, the increase in power is much higher than the gain in speed of the transistors.

The existence and identification of the minimum energy point has significance. It denotes the practical limits that can be attained by trading off performance to obtain energy savings. Any lowering of supply below that leads to sub-optimal operation in terms of both energy and performance.

In the subthreshold region of operation, namely from 400mV to 110mV, a performance range of 200 kHz to 270 Hz was observed. In this region, the energy consumed per operation ranged from 19 pJ/op to 6 pJ/op. In the superthreshold region of operation from 450mV to 700mV, which was the upper end of the examination range, the performance ranged between 500 kHz and 14.4 MHz. In this region, the energy consumed per operation ranged from 25 pJ/op and 110 pJ/op.

For circuits with tight power and energy budgets, the minimum energy point of operation and the available performance range in that vicinity are particularly important. Not all applications deployed using an FPGA tend to have identical energy and

performance budgets. It becomes important to measure the available latitude in terms of performance and energy. It can be seen the region from 200mV to 350mV offers a 30X performance improvement at just a 2X increase in energy, thus opening up to a larger class of applications and still within reasonable energy limits. In [8], the minimum energy point was reported at 350mV. Our results indicate the minimum energy point lies within deep-subthreshold. It is worth noting, that the minimum energy point of the individual blocks happens at around 150mV. When the circuit is implemented on a fabric, the minimum energy point is slightly offset towards 200mV. This is because, the fabric operating frequency is limited by the slowest of the component blocks and the critical path of the circuit.

4.5.2 Switching Influence

The data activity has a significant influence on the amount of dynamic power in Equation (2.10) and therefore affects the ratio of dynamic power to static power. It also estimates the ability of the circuit to respond to increased levels of input activity. In previous subthreshold works, this dependency was not considered. In this section, we examine this impact over a range of possible activity factors to assess its importance.

Figure 4.3 shows the impact of input switching activity on the energy over a range of supply voltages. The switching activity factors range from 25% to 100% of clock activity. At higher supply voltages, where total energy is dominated by the dynamic component, switching activity factor has an influence on the power-delay product. In subthreshold, where the total energy has less dependence on dynamic energy and is

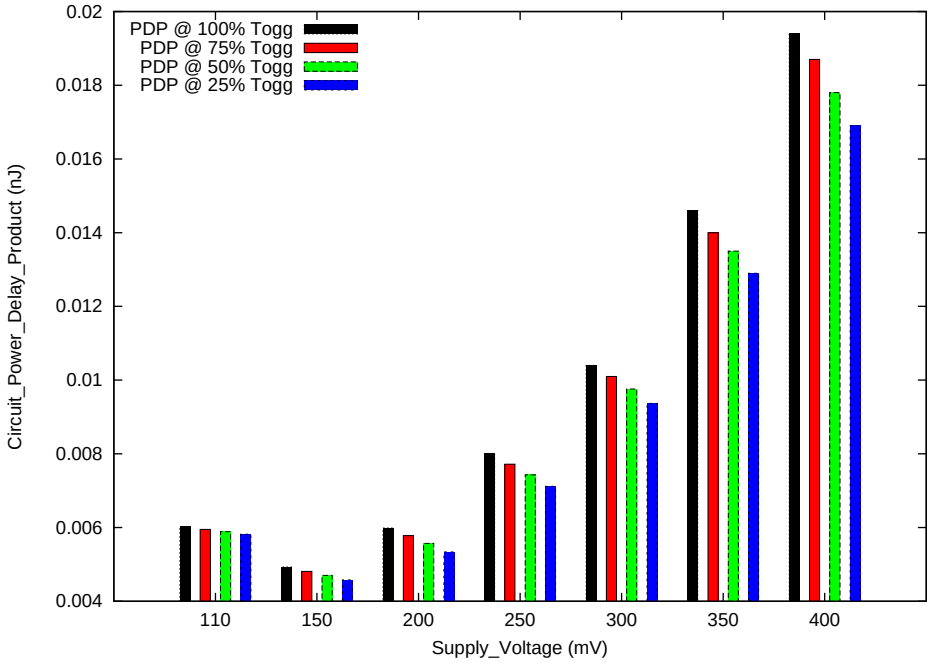

Figure 4.3: Input switching activity has a decreasing impact on energy in deep subthreshold.

influenced by static energy, switching activity factor does not have a major impact on the power-delay product. From this we conclude that input switching activity becomes less significant in deep subthreshold.

Methodologies like dynamic frequency scaling and dual-V_{dd} require additional circuitry and additional metal layers to provide on-demand power savings. It can be seen from Figure 4.3, such techniques are not very beneficial in subthreshold operation.

4.5.3 Energy Analysis

Total energy consumption depends on the energy consumption of the individual blocks. Determining the energy of the blocks can help FPGA synthesis tools meet target con-

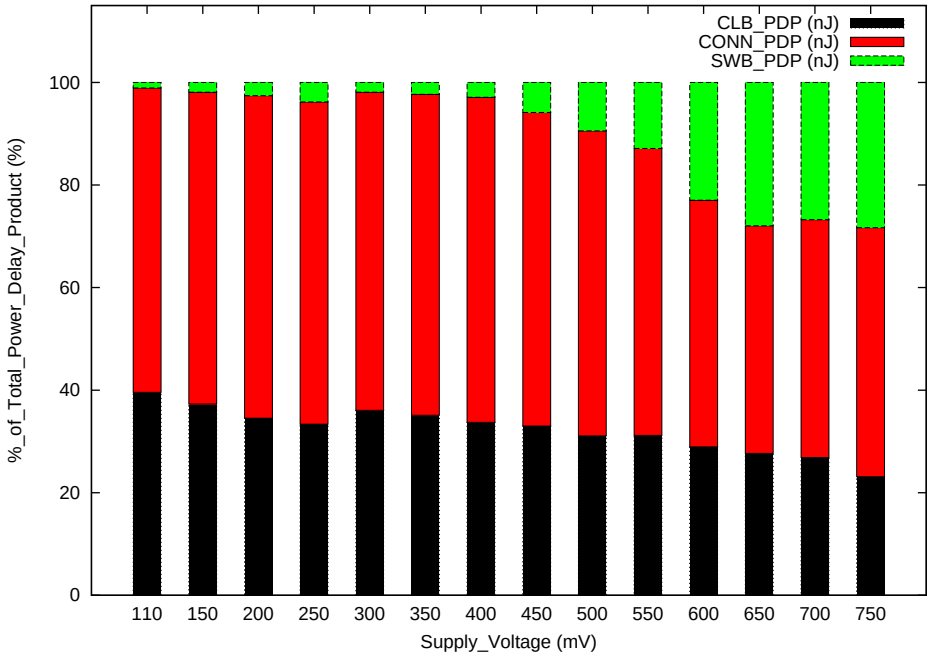

Figure 4.4: CLB energy increases in contribution to the total energy in deep subthreshold.

straints. At each supply voltage, we analyze the contribution of the different blocks to the total energy.

The bulk of the switching energy is consumed by CLB and CONN blocks according to Figure 4.4. This is consistent both at super-threshold and subthreshold. From subthreshold to super-threshold, however, the proportion of energy due to SWBs gradually grows.

The CLB and CONN blocks together contain more gates with power-ground paths, thus contributing to short circuit power. Also, these gates need to charge and discharge their load capacitances. Both of these operations involve some delay factor, which contributes to their larger share of energy consumption. In the case of SWBs

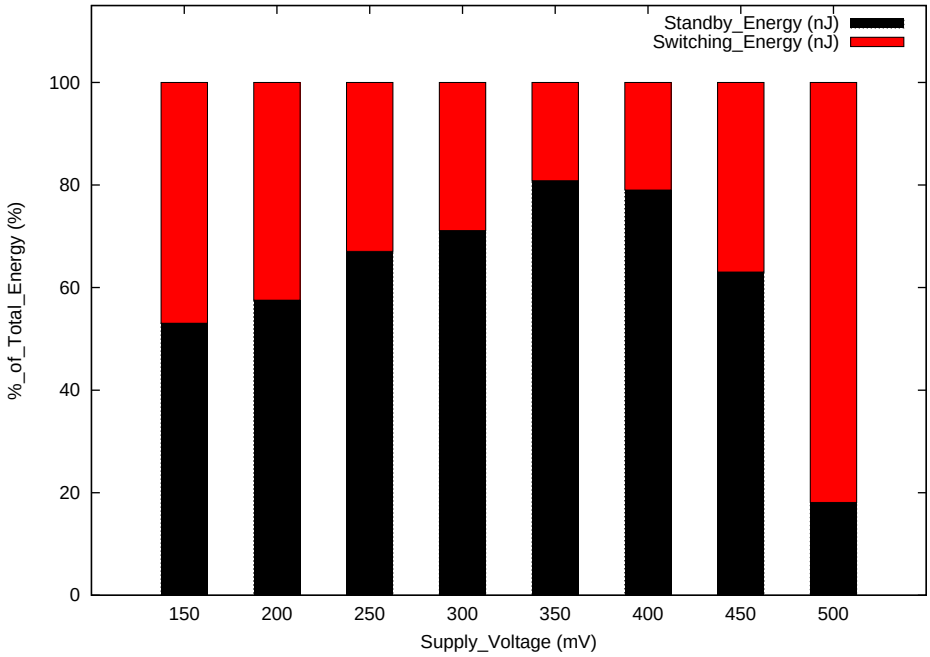

Figure 4.5: Switching energy has a less significant impact in near-threshold than standby energy, but again begins to increase in deep subthreshold.

which are made of transmission gates, each gate drives the source or drain of another gate and the load is comprised of only diffusion capacitance, which is usually less than the capacitance switched by CLB and CONN blocks.

Any architectural enhancements to CONN in an effort to reduce energy will necessitate corresponding modification to SWB and vice versa. So, their impact on energy cannot be estimated in isolation. However, since logic blocks by themselves account for about 40 % of overall energy architectural enhancements to them will provide direct energy savings.

4.5.4 Stand-by vs Switching Energy Analysis

Leakage due to transistors in the stand-by state is a significant source of energy consumption. In subthreshold operation, it is important to distinguish between the energy consumption due to active switching and that due to stand-by despite active switching actually using leakage currents.

We shutdown unused instances of the logic, connection and switch blocks when left unused by the place and route tool using a combination of power gating and clock gating. Figure 4.5 shows the contribution of stand-by to overall energy across a range of voltages. At subthreshold voltages, we found that stand-by energy dominates switching energy. On average, the stand-by energy in subthreshold region alone is about 67%. From subthreshold to super-threshold operation, the proportion of stand-by energy in total energy consumption reduces and the dominant component becomes switching energy which grows almost quadratically.

This is because, at subthreshold voltages, although the switching energy is low, the slow transistors spend more time in idle state and cause leakage. As the supply voltage scales up, the larger proportion of energy is due to active energy. The stronger currents flowing during the load capacitance charge and discharge processes and short circuit switching current cause this growing active energy. Although stand-by is unavoidable, it can be minimized by certain techniques such as self reverse bias (transistor stacks) and adaptive body bias.

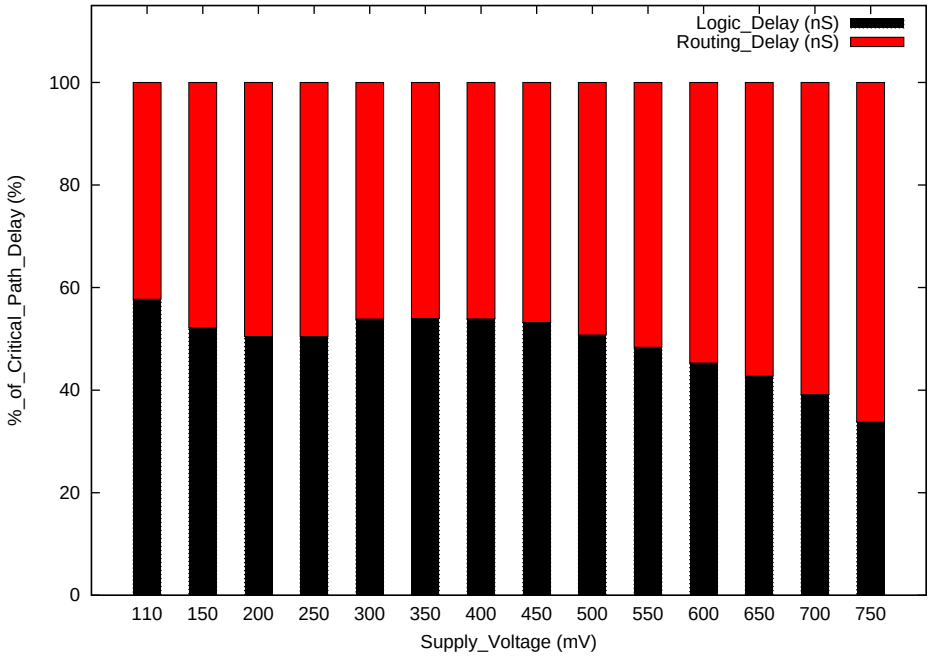

Figure 4.6: Routing delay has a decreasing impact on total delay in deep subthreshold.

4.5.5 Delay Analysis

Traditionally, the majority of the delay has been attributed to routing in FPGAs. It is very important to determine if this case holds true at subthreshold voltages or not. At each supply voltage, we analyze the contribution of logic delay and routing delay to overall circuit latency. This helps identify the primary cause of latency at each supply voltage.

Figure 4.6 shows that logic delay and routing delay are fairly equal at subthreshold voltages. When voltages are scaled down, the stand-by current through the gates of logic blocks flowing from VDD to GND is weak. The charging and discharging of load capacitances is very slow as it happens with these weak currents. Since the

CLBs possess more such gates in its path, the logic delay contribution increases and becomes comparable to routing delay.

In deep subthreshold routing delay becomes less significant and logic delay starts gaining importance. Architectural enhancements aimed at improving performance of FPGA should be directed at CLB design so as to obtain notable improvement in logic delay.

4.5.6 Energy sensitivity to Variation

The response of an 8-input multiplexer, which is a component of logic and routing blocks, to threshold voltage variation is discussed below. Table 4.1 shows the worst-case 3σ active and standby energy consumption of the representative 8-input multiplexer in the unbiased condition and when biased by a voltage at $V_{dd}/2$.

Table 4.1: Using a bias of $V_{dd}/2$ results in an increase in worst-case active and standby energy since the bias is not adaptive to the variation between transistors and logic gates.

Supply (mV)	Active Enegy (fJ)		Standby Energy (fJ)		Overhead (%)	
	Unbiased Circuit	Biased at $V_{dd}/2$	Unbiased Circuit	Biased at $V_{dd}/2$	Active Energy	Standby Energy
400	0.14	0.15	0.73	0.76	9.49	4.05
350	0.13	0.14	0.62	0.66	11.72	5.83
300	0.12	0.14	0.52	0.56	12.90	8.37
250	0.11	0.12	0.43	0.48	13.76	11.25

Threshold voltage variation can occur between the transistors in a logic gate, which affects the leakage currents and could cause timing and functional errors. The bias voltage delivers a current to both N-wells and P-wells to match the transistors and improve the performance. It should be noted, the bias is common to all the transistors in the circuit and does not distinguish the variation between logic gates. As a result,

depending on the variation, the bias could mitigate the variation or worsen it. However, a measure of the sensitivity of the circuit to variation, can lead to development of optimal bias techniques for circuits like FPGAs with regular structures.

From the simulation results, it can be seen, the circuits biased using the voltage $V_{dd}/2$ have a slightly higher worst-case energy across the range of supply voltages. This is observed in active energy and standby energy. The overhead at worst-case 3σ range between 4% to 13%, averaging 7.3% for standby energy, and 11.9% for active energy. The overhead increases with dereasing supply voltage. This is because, at the lower voltages, the impact of variation is so severe on the drain current mismatch between the FETs, and the applied bias at $V_{dd}/2$ is not sufficient to mitigate the variation.

The common bias to both wells is ineffective in mitigating worst-case energy variation, as it does not adapt to the variation. Biasing techniques that can adapt to the variation will instead be a better approach.

Chapter 5

Variation-aware Adaptive Body Biasing

Designing circuits for subthreshold operation is challenging because, the impact of process variation is more significant at reduced voltages due to the exponential dependence of drain current on gate voltage. Sensitivity to threshold voltage variation affects leakage, which is used for active circuit operation in subthreshold and during standby. Threshold voltage variation can occur in NMOS, PMOS devices and between these devices. When transistors in a logic gate are affected by variation, their drain currents and switching times are impacted, which could lead to functional and timing errors. For reliable timing and functionality in subthreshold operation, drain current variation caused by threshold voltage variation must be reduced.

Variation in analog circuits like current mirrors and voltage reference circuits is minimized by using large geometry devices. In subthreshold digital design, however, minimum sized devices are optimal for energy efficiency [12]. For a low target frequency, body biasing was more energy efficient than supply voltage scaling in mitigating process

and temperature variation [52]. Reverse bias was found to worsen drain current mismatch and forward bias reduced the mismatch [33]. Gate-level clustering with cluster specific body bias was found to improve leakage power [57].

Threshold voltage variation could affect transistor delays on critical paths, which in turn affects active energy, circuit performance and thus overall energy efficiency. To quantify the impact of variation on critical path delay, subthreshold design has used critical path replication [25] or typical approaches like block-based and path-based statistical analyses [7].

The third contribution of this thesis is a methodology to reduce worst-case standby energy caused by threshold voltage variation. A design methodology is proposed to use forward bias with a body bias regulator [2]. The previous regulator is used but a digital design methodology was developed to mitigate the impact of variation. This chapter presents the methodology of finding the optimal number of regulators, determining their placement and assigning cells to be biased by these regulators.

5.1 Overview of methodology

Consider an inverter with equal rise and fall times, biased by an equal sized regulator as shown in Figure 5.1. The regulator is designed to output a voltage of $V_{dd}/2$ since the two FETs in cutoff act like a voltage divider. In the presence of threshold voltage variation, the actual regulator output voltage will be higher or lower than $V_{dd}/2$ depending on the specific FET threshold voltages.

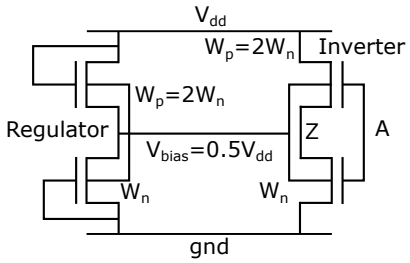

Figure 5.1: An inverter biased by the bias regulator circuit when the transistors are perfectly matched.

The threshold voltages of NFETs (and PFETs) of both circuits are assumed to be normally distributed around the nominal value. According to Pelgrom's model the variance of the parameter mismatch between transistors is directly proportional to the distance between them and inversely proportional to the area of the devices [34]. Given the distance between two transistors, variance of their threshold voltage mismatch can be computed using Pelgrom's model. Using variance of mismatch, variance of threshold voltage distribution and uncorrelated random samples from this distribution, correlation between random variables is determined.

In Figure 5.2, the worst-case 3σ standby energy of an inverter biased by the regulator and an inverter biased by an off-chip voltage source at $V_{dd}/2$, are compared. In the case of inverter biased by the regulator, it can be seen, the inverter worst-case standby energy decreases over a range of correlation values which changes over distance between inverter and regulator.

When a circuit is biased by an off-chip voltage to mitigate variation, meeting a target performance or a leakage constraint, requires tuning the well biases individually in addition to maintaining an offset to minimize device mismatch. Bias and offset values

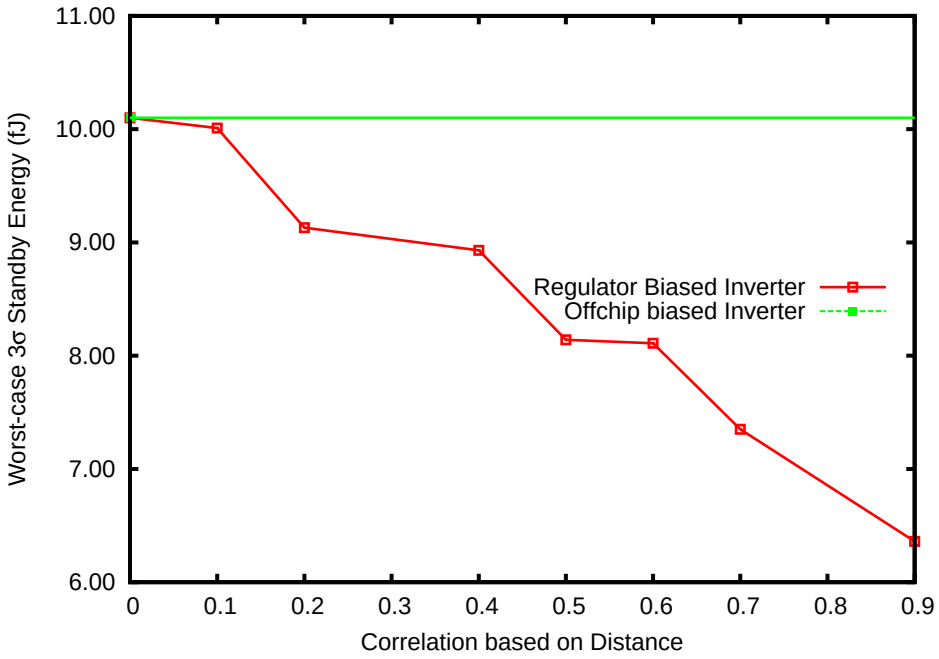

Figure 5.2: On-chip bias regulators improve worst-case 3σ standby energy better when they are nearby, and hence more correlated, with the circuit they bias.

for target performance can be determined by at-speed testing of silicon. However, even with a few bias assignment values this testing is time consuming and requires a range of bias voltage generators. In the case of leakage constraints, specialized circuitry like reference mirrors to measure current mismatch and pin overhead to measure leakage are required. Instead, a simple way to mitigate impact of variation is biasing both wells by the same amount. For this work, $V_{dd}/2$ is used, because, the substrate current is much smaller than drain current and thus scalable to deep subthreshold voltages.

5.2 On-chip Regulator Design Methodology

Standard cell design flows use a set of pre-designed library of cells. From Figure 5.2, savings in worst-case 3σ energy can be observed when the regulator is nearby and thus correlates better with the cell. The regulator circuit consumes standby power and incurs area. This methodology makes a trade-off in the number of regulators and the performance of the circuit. Filler cell locations are targeted for regulator insertion. This methodology is compatible with standard tool flows and can be deployed on existing designs with minimal overhead.

5.2.1 Regulator Design

This work uses the Nangate Open Cell Library implemented using the 45nm FreePDK process technology [44], [45]. The library cells were sized for equal rise and fall times in the worst case switching condition. The impact of variation can vary significantly between a library cell and its associated regulator, which in turn could affect the leakage current match and hence energy savings. For improved matching, the transistor dimensions of the regulator need be comparably similar to that of the library cells. So, a set of regulators were custom designed based on the functionality and transistor dimensions of the library cells.

The performance of the cells biased by equal sized regulators were compared with cells biased by minimum sized regulators. In the case of NAND gates, it can be observed, the equal sized regulators provided more energy savings than minimum sized

regulators. This is because, the series transistors in NMOS network are sensitive to improved current matching and stack effect, which in turn gives energy savings. In the case of NOR gates, the minimum sized regulators performed better. This is because the leakier NMOS transistors are in parallel, with no stack effect and thus, the equal sized regulators constitute an energy overhead. From this it can be concluded, using cell specific regulators offers energy savings.

Cell bias pins access the wells depending on the implementation of the library by the foundry, either tapped or tapless. Tapped cells have a tap contact reaching into the wells which is connected to the output net of the regulator. On the other hand, tapless cells require an additional tap cell with its output connected to the net of the regulator. While this incurs an implementation complexity, there is no power overhead as the tapcells do not have any transistors.

5.2.2 Cell Characterization

The worst-case standby energy of the inverter follows a linear trend assuming correlation changes over distance between inverter and regulator. In addition, this energy also depends on the circuit topology and cell functionality. So, to use this methodology in a standard cell based flow targeting worst-case energy savings, an accurate estimation of the energy savings of the cells in the library is needed. All cells in the Nangate Open Cell Library were characterized for power and performance using cell specific regulators and with the variation model that was used to characterize the inverter circuit described earlier.

Using the characterization data, linear models were fit for the worst-case standby energy savings of each cell in the library, as a function of distance. For instance, consider 2-input NAND and NOR gates biased by equal sized regulators, compared with gates biased by minimum sized regulators. The energy savings of each of these varies depending on the circuit topology, number of transistor stacks and functionality. Figure 5.3 shows the linear functions for these three gates.

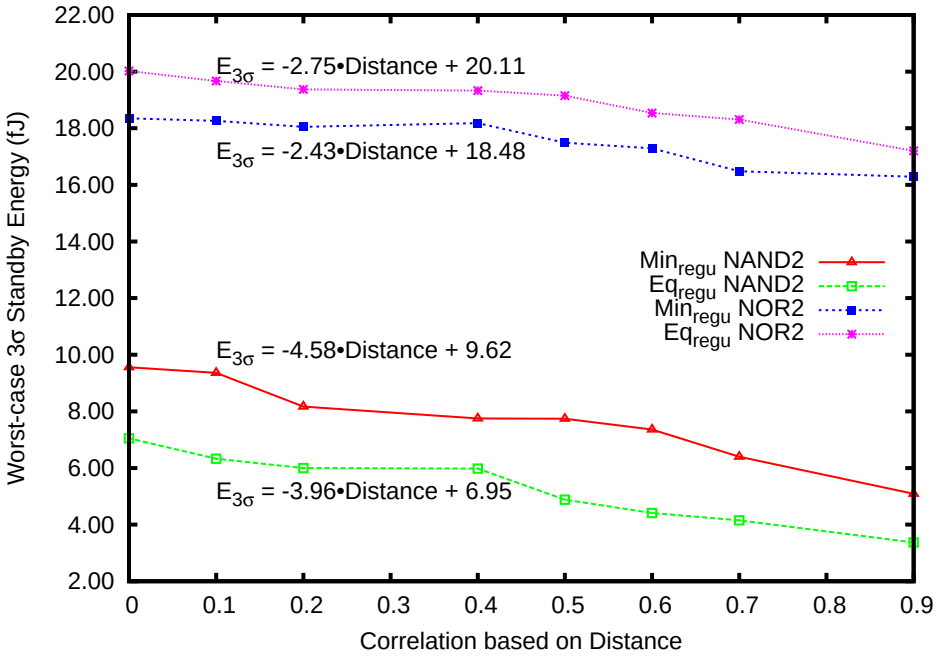

Figure 5.3: Energy savings for NAND2 and NOR2 gates biased by specific on-chip regulators can be modeled as a linear function.

5.2.3 Design Implementation

A CAD flow with industry standard tools to optimally assign regulators to groups of standard cells, was developed as a part of this work. The regulator assignment is

formulated as a Linear Program (LP), which is described in the next section.

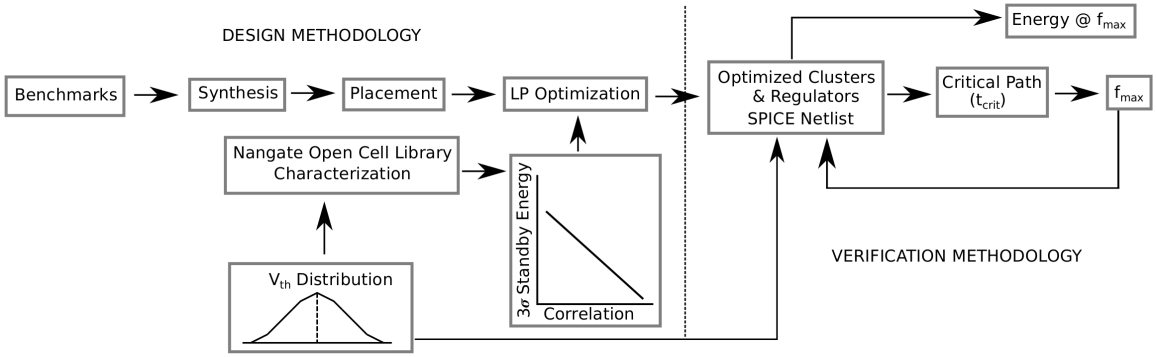

Figure 5.4: Block diagram showing on-chip bias regulator design and verification methodology.

Figure 5.4 shows the CAD flow of the proposed design and verification methodology. Starting with Verilog description of the benchmarks, were synthesized using Synopsys Design Compiler and Nangate Open Cell Library to obtain gate level netlists [**?**]. These netlists were placed using Synopsys IC Compiler. Upon completion of placement phase, physical implementation of the circuit contains filler cells in the unused areas. The LP optimizer reads in the placed netlists containing geometric coordinates of the cell and filler instances. Because the regulators consume standby power, there is no need to replace all of the filler cells, but they are all candidate locations for regulator instances. The optimizer solves for the optimal clustering of cells biased by regulators, using LP_Solve [32]. In the physical implementation, adjacent rows have shared wells. In addition, for each cluster of cells biased by a regulator, by defining the bias regulator output as a power pin, and by defining the cluster as a voltage domain, routing can be accomplished using industry standard tools.

Figure 5.5 shows a representative optimization of regulators connected to stan-

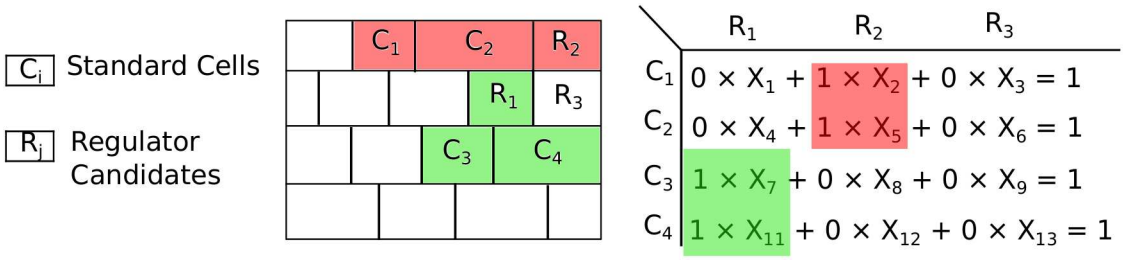

Figure 5.5: A set of cells and regulators are mapped to an application of LP constrainted optimization as in Eq. 3 and 4.

dard cells. To physically implement this aspect of the methodology, features available in typical CAD tools were leveraged. Relative placement groups for each cluster of cells connected to a regulator were created and a power domain associated with each group. The output net of each regulator was designated as a supply net and those nets were explicitly connected to the cell bias pins.

5.2.4 Formulation

Insertion of each regulator incurs a standby energy cost. So, to achieve worst-case energy savings at the circuit level the optimal number of regulators, the placement of these regulators and which cell(s) are assigned to a regulator, are all determined. The regulators take the place of filler cells and bias the standard cells in the vicinity. It should be noted, the energy savings of a standard cell biased by a regulator is determined by the topology of the cell and the distance from its regulator.

The problem of assigning regulators to one or more groups of standard cells is modeled as an LP. Two algorithms to solve this LP problem are demonstrated. The first is an exhaustive approach yielding the optimal solution, while the second is a

faster, heuristic solution. The two approaches differ in the number of LP constraints formulated and how they generate the constraints. In the optimal solution, all filler cells are considered as candidate locations for regulators, each of which can be paired with any cell on chip. In the heuristic approach, subsets of filler cell locations are derived from the set of all filler cell locations, as candidates for regulator insertion. The regulators to make clusters with cell instances are then determined.

For these subsets, only rows immediately adjacent to a cell are considered, for each cell in the design. Using this row-step, constraints are built to attempt to solve the LP. If the LP does not converge, the set of rows from which the subset of filler cells are derived, is increased. This is done for each cell in the design. The LP solution is attempted again.

The goal of the heuristic is to determine the smallest number of rows from which the subsets can be derived leading to a solvable LP. The goal of the LP formulation is to identify a subset of regulators from the available regulators and determine the grouping of standard cells to be assigned to those selected regulators. The LP formulations of both algorithms have identical cost functions and differ only in the number of constraints. So, for brevity, only the equations of the optimal formulation are presented. The areas where the formulations differ are indicated.

- Let m represent the number of cell instances in the design.
- Let i be an index variable such that $i = 0,1,.....(m\text{-}1)$.
- Let C represent the set of all cells indicated by C_i where $i = 0,1,...(m\text{-}1)$.

- Let n represent the number of regulators in the design.
- Let j be an index variable such that j = *0,1,.....(n-1).*
- Let R represent the set of regulators indicated by R_j where j = *0,1,...(n-1).*
- Let a be a constant representing the standby energy cost of a single regulator.
- Let X_{ij} represent a decision variable taking values $\{0, 1\}$ indicating whether a specific cell C_i is assigned to a specific regulator R_j.
- Let Y_i represent an auxiliary variable denoting the cost of a specific cell being assigned a specific regulator from the available regulators.
- Let e_{ij} represent the energy savings coefficient for a given cell C_i regulator R_i assignment. This is obtained from the linear energy savings model discussed earlier, with distance of a fixed regulator to a given cell as input to this model.

The LP solver solves for decision variables X_{ij} which determines the optimal grouping of cells biased by regulators.

The formulation is described below:

$$\text{Minimize} \sum_{i=1}^{m} Y_i + m \cdot a \quad \text{such that} \tag{5.1}$$

$$\sum_{j=1}^{n} [e_{(i \cdot n)+j} \cdot X_{(i \cdot n)+j}] \leq Y_{(i+1)} \;\; \forall i = 0, 1, 2...(m-1) \tag{5.2}$$

$$\sum_{j=1}^{n} X_{(i \cdot n)+j} \;\; = \;\; 1 \quad \forall i = 0, 1, 2...(m-1) \tag{5.3}$$

Algorithm 1: Algorithm finds optimal cell clusters and assigns bias regulators to each of these clusters

1 function optimal_bias ;

Define : RSS = Row Search Space,

RSS_{max} =Number of rows in placed chip

Input : Placed File containing cells and fillers,

Cell-specific characterizing information

Output: Gate clustering,

Optimal regulator assignment

2 write ILP constraints using RSS_{max};

3 solve_ILP() ;

Algorithm 2: Heuristic algorithm finds clusters of cells and assigns bias regulators to each of these clusters

1 function heuristic_bias ;

Define : RSS = Row Search Space,

$RSS_{min} = 1$,

RSS_{max} =Number of rows in placed chip,

$RSS_{curr} = RSS_{min}$

Input : Placed File containing cells and fillers,

Cell-specific characterizing information

Output: Gate clustering,

Regulator assignment

2 **while** $RSS_{curr} \leq RSS_{max}$ **do**

3 write ILP constraints using RSS_{curr};

4 solve_ILP() ;

5 $RSS_{curr}++$

6 **end**

$$X_p = \{0, 1\} \quad \forall p = 1, 2, ...(m \cdot n) \tag{5.4}$$

The cost function in Equation 5.1 describes the goal of this formulation, which is to minimize the number of clusters into which all standard cells can be grouped, such that each cluster is connected to a regulator. Here Y_i denotes the linearized worst-case 3σ standby energy cost of a cell i from amongst the set of all cell instances C, when biased by a regulator. Summed over the set of all cell instances, the goal is to minimize the cost of clustering all the cell instances in the design. The second term indicates the standby energy cost of the regulators needed to cluster all the cell instances.

The set of constraints denoted by Equation 5.2 determine which of the cells get clustered together to be biased by a common regulator and get assigned a regulator. The coefficient term is the energy cost of driving a particular cell with a particular regulator. This term is obtained by precharacterizing the cells of the standard cell library and the distance between the regulator and the cell.

The LHS of constraint Equation 5.2 represents the energy cost of biasing a given standard cell by each of the available regulators. The RHS of constraint Equation 5.2 ensures this cell is biased by one of the available regulators only.

The constraint Equation 5.3 ensures each cell has to be driven by a regulator and enables grouping of cells to a common regulator. The constraint Equation 5.4 indicates X_{ij} is a binary decision variable taking values {0,1}.

The above described constraints apply to the Algorithm-1. For Algorithm-2, the indices of the variables and the limits of the summation are not a constant n and instead take variable values based on the number of regulator candidates present in the

rows adjacent to each cell.

Regulator insertion matches the transistor off-current using localized and better correlated regulators. This enables improved performance, active energy and standby energy in the worst-case. The goal is to minimize the linearized worst-case energy cost of each cell in the design by assigning regulators and sharing the regulators with cell clusters in the absence of candidate regulators. The formulation does not restrict regulator deployment directly. In the presence of candidate regulators, near a cell, each regulator offers a linearized savings in comparison against a single hypothetical regulator. A single hypothetical regulator for the entire design, will offer linearized savings to cells only in its vicinity. For cells, elsewhere, other candidate regulators in their vicinity offer better savings, causing the optimizer to select them, as opposed to the single regulator. Off-current matching provided by the regulators is the mechanism using which the worst-case reduction of active and standby energy of the cells is achieved. Optimizing the number of regulators (between a minimum of 1 and maximum of 1 per cell) limits the standby energy and area overhead cause by the regulators. This methodology targets savings in both active and standby energy and not directly total energy.

5.3 Experimental Methods

The Nangate Open Cell Library implemented using the 45nm FreePDK process is used as the standard cell library. The bias regulator was designed using this process. The nominal threshold voltages of the transistor models in this process were $V_{th0_N} = 0.4106$

V and $|V_{th0_P}| = 0.3842$ V.

5.3.1 Variation Model

Parametric variation of threshold voltage caused by Random Dopant Fluctuation (RDF) is considered in this work. The input variation model is based on Pelgrom's model given by,

$$\sigma^2 (V_{th}) = (S^2_{V_{th}} \cdot Distance^2) + (\frac{A^2_{V_{th}}}{W \cdot L}) \cdot \tag{5.5}$$

$S^2_{V_{th}}$ and $A^2_{V_{th}}$ are technology specific constants in the range of 0.01 mV per micrometer and 0.001 mV respectively. Using these fitting constants, the correlation drops to zero at a distance of 8 micrometer. he threshold voltage is assumed to be normally distributed around the nominal value, with 1σ variance of 20mV [65]. All instances of standard cells and regulators are subject to threshold voltage variation from the distribution. Given spatial separation between a cell instance and regulator instance and the variance of mismatch between them, the respective threshold voltages of the FETs are determined. For a given mismatch variance, which is a function of the spatial separation, and the threshold voltage distribution, the correlation between the random variables is computed. Two randomly picked threshold voltage values are transformed into correlated values using the computed correlation coefficient, by applying Cholesky decomposition [47].

5.3.2 Optimized Circuit

The optimizer reads in placed netlists containing geometric coordinates of the cell and filler instances. Considering filler cell instances as candidate locations for regulator insertion, the optimizer solves for the optimal clustering of cells biased by regulators and the locations of regulators, using LP_Solve [32]. For clusters containing two or more cells, the regulators assigned could potentially belong to one of many types, corresponding to the regulator-types best suited for each cell-type. Assigning a large sized regulator, best suited to one of the cells in the cluster, will render the regulator less representative to other smaller sized cells in the same cluster. So, a minimum sized regulator is assigned to such clusters, to keep regulator area overhead minimum and to estimate worst-case effectiveness of regulator methodology. Based on cell level characterization, this could result in pessimistic estimate of 5% - 24% across the cells in the library. This was obtained by analyzing the difference in savings accrued by using cell-specific regulators against minimum sized regulators in the characterization. The regulator bias lines do not have switching activity. So, their resistance is not considered. Since the wells have very low current requirement, the IR-drop is negligible. Using the results of optimization and placed netlists, SPICE netlists containing clusters of cells biased by their regulators, are obtained. The variation model discussed earlier, is applied on these netlists.

5.3.3 Static Timing Analysis

Circuit timing paths are affected when threshold voltage variation causes delay variation in transistors. As a result, a non-critical path could become critical or vice-versa. Either way, this affects target performance of the circuit. This variation in critical path delay in turn affects the circuit power consumption and hence energy. Since subthreshold circuits are highly sensitive to variation, even small approximations in critical path delay can have a significant impact on performance and energy. So, an accurate estimation of the critical path delay is required. The exact critical path when variation is applied to the circuit, is determined. The circuit elements constituting the critical path varies depending on variation. The critical path is identified by performing transistor level static timing analysis using Synopsys Nanotime. Using critical path delay, the maximum operating frequency f_{max} of the circuit is determined.

5.3.4 Energy Measurement

Using critical path delay, the circuit is simulated at f_{max} using random input stimuli. The active and standby energy per cycle are measured using Synopsys Hspice.

The above process of applying variation, determining critical path, finding f_{max} and using it to compute energy is performed for 1000 iterations of Monte Carlo simulation. The benchmark performance is then compared with the unbiased circuit and a circuit biased using a voltage source at $V_{dd}/2$, described earlier in Section , comparable to an off-chip source. Since the regulators are connected to on-chip V_{dd}, these results

include the energy overhead of the regulators.

5.3.5 Impact on area

Unused filler cells are used as candidates for regulator assignment. This eliminates the need to create spaces for the regulators. However, each regulator could output a bias voltage different from another regulator, necessitating design rule spacing for wells at different potentials. The spacing overhead between clusters is measured using this design rule constraint.

5.4 Results

This section presents the results of the optimization process and the impact of optimization driven body bias on energy and performance of ISCAS85 benchmarks.

Table 5.1 shows the run time of the optimization process for several benchmarks using the optimal and heuristic approaches. The improvement in run-time of the heuristic solution is due to the reduced subset of candidate regulators considered for each cell by the optimizer.

Table 5.1: Optimization run-times for ISCAS85 benchmarks using Optimal and Heuristic solutions

Circuit	Optimal (Sec)	Heuristic (Sec)	% Improvement
c432	1	1	0.0
c499	12	7	41.6
c1355	12	8	33.3
c1908	15	10	33.3
c2670	19	12	36.8
c3540	66	43	34.8
c5315	229	160	30.1

Improvement in energy and performance is evaluated at the extremeties of the resulting distributions. The active energy, standby energy and delay exhibited log-normal distributions. So, the distributions were evaluated using typical parameters namely mean (μ) and worst-case 3σ. The regulator bias method provides improvement by delivering a forward bias which is adapted to the cell variation in that spatial vicinity. This locally relevant forward bias offer a better matching of the FET off-currents and lowers the cell threshold voltage, thus improving its performance. This increased performance enables scaling the circuit to lowered supply voltages, thus saving worst-case active energy and standby energy.

Inter-well spacing for wells at different potential was considered to measure the impact on circuit area. This area overhead spans a range of 17.5% at the minimum to 24.5% at the maximum across all benchmarks with an average of 19.4%.

From Figure 5.6, it can be observed, the regulator method offers improvement in worst-case delay of the circuit for all benchmarks over a range of supply voltages compared to an unbiased circuit. The improvements decrease as the supply voltage is scaled down. This is because, at lower operating voltages, the impact of variation is much higher than the applied bias compensation. The improvement in delay spans a range of 10.89% at the minimum to 50.16% at the maximum. Considering all benchmarks, the average improvement in worst-case delay is 36.93%, 27.85% and 12.74% at 350mV, 300mV and 250mV respectively using Algorithm-2, compared with an unbiased circuit.

From Figure 5.7, it can be observed, the regulator method offers savings in

Table 5.2: Savings in worst-case 3σ standby energy using on-chip regulator assignment at V_{dd}=350mV

Circuit	Unbiased vs On-chip biased					Offchip biased vs On-chip biased				
	Unbiased (fJ)	Algo-1 (fJ)	Savings (%)	Algo-2 (fJ)	Savings (%)	Offchip bias (fJ)	Algo-1 (fJ)	Savings (%)	Algo-2 (fJ)	Savings (%)
c432	0.92	0.60	41.95	0.87	5.70	1.04	0.60	42.30	0.87	17.12
c499	1.15	0.98	1.64	1.24	-8.13	1.32	0.98	25.75	1.24	6.41
c1355	1.16	0.90	3.30	1.26	-8.61	1.29	0.90	30.23	1.26	2.69
c1908	1.67	1.35	12.52	1.65	1.41	1.74	1.35	22.41	1.65	5.50
c2670	2.00	1.96	4.07	1.98	1.15	2.04	1.96	3.92	1.98	3.06
c3540	1.94	1.80	5.21	1.95	-0.82	2.04	1.80	11.76	1.95	4.03
c5315	2.34	2.16	3.15	2.36	-0.57	2.43	2.16	11.11	2.36	2.97
Avg.			10.26		-1.41			21.06		5.96

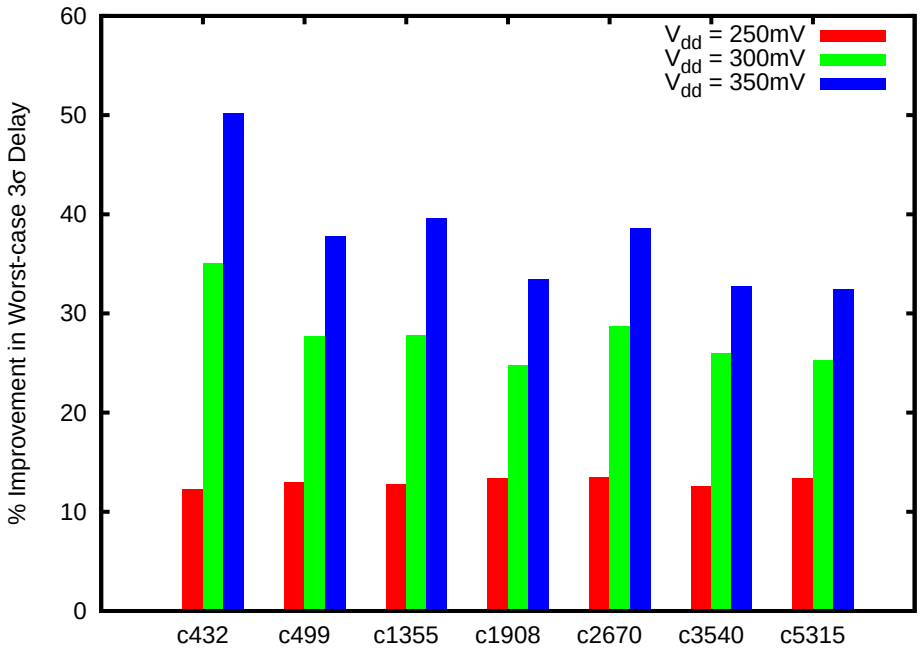

Figure 5.6: On-chip regulator methodology improves worst-case 3σ delay compared to an unbiased circuit over a range of supply voltages.

worst-case active energy of the circuit for all benchmarks when compared against an unbiased circuit and an offchip biased circuit. This was verified using both algorithms. The active energy savings on average compared to an unbiased circuit across all benchmarks include 14.52% and 4.50% for Algorithm-1 and Algorithm-2 respectively. Compared to an offchip biased circuit, the savings are 18.84% and 9.20% using Algorithm-1 and Algorithm-2 respectively. Algorithm-1 slightly outperforms Algorithm-2 in both cases of comparison, namely against an unbiased circuit and offchip biased circuit. This is because, the optimal solution algorithm offers a wider choice of regulators to choose from, resulting in improved savings.

In Table 5.2 we list the savings in worst-case 3σ standby energy when using

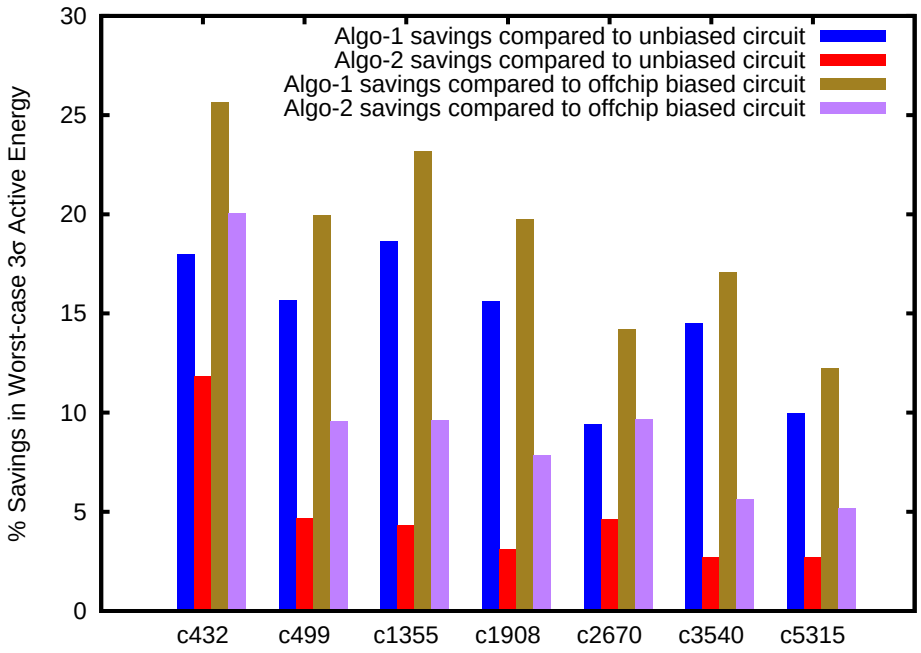

Figure 5.7: On-chip regulator methodology improves worst-case 3σ active energy compared to an unbiased circuit and offchip biased circuit using V_{dd}=350mV

the on-chip regulator methodology. This is compared against unbiased circuits and with offchip biased circuits using both algorithms. The standby energy savings on average compared to an unbiased circuit across all benchmarks include 10.26% and -1.41% for Algorithm-1 and Algorithm-2 respectively. Compared to an offchip biased circuit, the savings are 21.06% and 5.96% using Algorithm-1 and Algorithm-2 respectively.

Compared to an offchip biased circuit, regulator assignment using both algorithms offers standby energy savings. Compared to an unbiased circuit, Algorithm-1 offers savings across all benchmarks, while Algorithm-2 offers savings in some cases. In other cases there is a slight increase in the worst-case standby energy. This corresponds to the cases where the active energy savings are also at the lowest, due to the reduced

choice of regulators available. At this point, the regulators become an overhead. Benchmark c7552 was computationally intensive using Algorithm-1 and so was analyzed using Algorithm-2 only.

The goal of the heuristic is to determine the smallest number of rows from which a subset of regulators can be derived leading to a solvable LP. In this work, this was accomplished using 1-row. Compared to the optimal algorithm, the heuristic algorithm provides optimization speed-up at the expense of considering fewer candidate regulators, and so, reduced energy savings. The reduced energy savings was observed in the optimization process as a higher cost function, compared to the cost function yielded by the optimal algorithm. Applying cell-specific regulator assignment to single-cell clusters and not for other clusters is a limitation to this methodology. This leads to divergence between modeled and simulated energy. At the circuit level, this error could be as high as 42% and in some cases around 20%. A more accurate modeling and implementation by characterizing the regulator shared capacity and additional regulator types for select typical clusters would have yielded closer correlation between modeled energy and simulated energy. This was not considered in the optimization assuming cell clustering is circuit and implementation-specific. From a well-access perspective, cells that are not adjacent pose a physical complexity, in addition to the reduced energy savings. This is a limitation to the methodology. Physically-aware constraints can overcome this limitation. Considering only directly adjacent cells and regulators will provide better correlation and hence improved energy savings. In addition, it will offer improved physical implementation.

Chapter 6

Conclusion and Future Work

Continued leverage of subthreshold operation for energy constrained applications is possible only if impact of variation is minimized. Whether using custom-design methodology or FPGA methodology, the impact of variation on power is significant enough to reduce the intended power savings. This thesis studied the performance of circuits in subthreshold to determine the constituent factors to circuit energy and delay. Circuit sensitivity to process variation was evaluated and mitigation techniques to reduce worst-case energy and performance were proposed.

6.1 Thesis Contributions

The contributions of this thesis are:

- Design and implementation of a single-V_{dd} subthreshold FPGA with body bias methodology to mitigate the impact of process variation. This included the circuit level cell library design and design automation flow required to implement the

FPGA fabric for manufacturing.

- Determination of energy optimality in subthreshold FPGAs and development of a methodology for this using a simulation based characterization framework [48]. This framework can be applied to evaluate new FPGA fabric implementations.

- Development of a variation model and application of it to benchmarks to measure variation in subthreshold circuits. The underlying infrastructure overcomes the limitation of typical transistor level simulation engines by applying correlated variation.

- Development of a methodology to mitigate variation in subthreshold circuits. This methodology used forward body bias as compensation mechanism to reduce the impact of variation [49].

6.2 Future Work

Timing measurement is vital in estimating energy of subthreshold circuits. Development of transistor level timing analysis tools considering both subthreshold and normal operation could be a useful future work with influence on any subthreshold research.

Development of FPGA CAD tools with the ability to account for body bias and even parametric variation will be a useful addition to the existing tools.

Development of area efficient reverse bias voltage generators and logic circuitry to perform timed-delivery for extended periods of circuit inactivity. This could be an extension to one of the contributions of this thesis with wider application scope.

Bibliography

[1] W. Lee, P. Landman, B. Barton, S. Abiko, H. Takahashi, H. Mizuno, S. Muramatsu, K. Tashiro, M. Fusumada, L. Pham, F. Boutaud, E. Ego, G. Gallo, H. Tran, C. Lemonds, A. Shih, M. Nandakumar, B. Eklund and I. Chen. A 1 V DSP for Wireless Communications. *Proceedings of the International Solid State Circuits Conference*, pages 92–93, Feb 1997.

[2] A. Bryant, J. Brown, P. Cottrell, M. Ketchen, J. Ellis-Monaghan and E.J. Nowak. Low Power CMOS at Vdd = 4kT/q. In *Proceedings of the Device Research Conference*, pages 22–23, June 2001.

[3] A. Chandrakasan, W. Bowhill and F. Fox. *Design of High-Performance Microprocessor Circuits*. IEEE Press, 2001.

[4] A. Keshavarzi, S. Ma, S. Narendra, B. Bloechel, K. Mistry, T. Ghani, S. Borkar and V. De. Effectiveness of Reverse Body Bias for Leakage Control in Scaled Dual-Vt CMOS ICs. In *Proceedings of the International Symposium on Low Power Electronics and Design*, pages 207–212, August 2001.

[5] A. Rahman and V. Polavarapuv. Evaluation of Low-Leakage Design Techniques for Field Programmable Gate Arrays. *Proceedings of International Conference on FPGAs*, pages 23–30, February 2004.

[6] A. Raychowdhury, B. Paul, S. Bhunia and K. Roy. Computing With Sub-threshold Leakage: Device Circuit Architecture Codesign for Ultralow Power Subthreshold Operation. *IEEE Transactions on VLSI*, 13(11):1213–1224, 2005.

[7] A. Srivastava, D. Sylvester and D. Blaauw. *Statistical Analysis and Optimization for VLSI: Timing and Power*. Springer, 2005.

[8] A. Wang and A. Chandrakasan. A 180mV Subthreshold FFT processor. *IEEE Journal of Solid State Circuits*, 40:310–319, 2005.

[9] Actel Corporation. Accelerator Series FPGAs - ACT3 Family. *http://www.microsemi.com/document-portal/doc-view/130668-accelerator-series-fpgas-act-3-family*.

[10] Actel Corporation. High Performance FPGAs: SX-A Family. *http://www.microsemi.com/document-portal/doc-view/130722-sx-a-family-fpgas-datasheet*.

[11] B. Calhoun, A. Wang and A. Chandrakasan. Modeling and Sizing for Minimum Energy Operation in Subthreshold Circuits. *IEEE Journal of Solid State Circuits*, pages 1778–1786, 2005.

[12] B. Calhoun, A. Wang and A. Chandrakasan. Device Sizing for Minimum Energy Operation in Subthreshold Circuits. In *Proceedings of the Custom Integrated Circuits Conference*, pages 95–98, March 2007.

[13] B. Zhai, D. Blaauw, D. Sylvester and K. Flautner. Theoretical and Practical Limits of Voltage Scaling. In *Proceedings of the Design Automation Conference*, pages 868–873, June 2004.

[14] B. Zhai, S. Hanson, D. Blaauw and D. Sylvester. Analysis and Mitigation of Variability in Subthreshold Design. In *Proceedings of the International Symposium on Low Power Electronics and Design*, pages 20–25, Aug 2005.

[15] B. Zhai, S. Hanson, D. Blaauw and D. Sylvester. A Variation-Tolerant Sub-200mV 6-T Subthreshold SRAM. *IEEE Journal of Solid State Circuits*, 43(10):2338–2348, October 2008.

[16] V. Betz and J. Rose. VPR: A New Packing, Placement and Routing Tool for FPGA Research. *Proceedings of the International Workshop on Field-Programmable Logic*, pages 213–222, Aug 1997.

[17] C. Neau and K. Roy. Optimal Body Bias Selection for Leakage Improvement and Process Compensation Over Different Technology Generations. In *Proceedings of the International Symposium on Low Power Electronics and Design*, pages 116–121, August 2003.

[18] D. Coue and G. Wilson. A Four-Quadrant Subthreshold Mode Multiplier for Analog

Neural-Network Applications. *IEEE Transactions on Neural Networks*, 7(5):1212–1219, September 1996.

[19] E. Vittoz and J. Fellrath. CMOS Analog Integrated Circuits based on Weak Inversion Operation. *IEEE Journal of Solid State Circuits*, pages 224–231, Jun 1977.

[20] R. Gonzalez, B. Gordon, and M. Horowitz. Supply and Threshold Voltage Scaling for Low Power CMOS. *IEEE Journal of Solid State Circuits*, 32:1210–1216, Aug 1997.

[21] P. Grossman, M. Leeser, and M. Onabajo. Minimum Energy Analysis and Experimental Verification of a Latch-Based Subthreshold FPGA. *IEEE Transactions on Circuits and Systems II: Express Briefs*, pages 942–946, Dec 2012.

[22] H. Soeleman and K. Roy. Ultra-low Power Digital Subthreshold Logic Circuits. *Proceedings of International Symposium on Low Power Electronics and Design*, pages 94–96, Aug 1999.

[23] H. Soeleman and K. Roy. Digital CMOS Logic Operation in the Sub-Threshold Region. *Proceedings of IEEE Great Lakes Symposium on VLSI*, pages 60–65, Mar 2000.

[24] H. Hassan, M. Anis, and M. Elmasry. Input Vector Reordering for Leakage Power Reduction in FPGAs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 155–1564, Sep 2008.

[25] I. Chang, S. Park and K. Roy. Exploring Asynchronous Design Techniques for

Process-Tolerant and Energy-Efficient Subthreshold Operation. *IEEE Journal of Solid State Circuits*, 45(2):401–410, February 2010.

[26] J. Cong and Y. Ding. FlowMap: An Optimal Technology Mapping Algorithm for Delay Optimization in Lookup-Table based FPGA Designs. *IEEE Transactions on Computer Aided Design*, pages 1–12, Jan 1994.

[27] J. Kwong and A. Chandrakasan. Variation-Driven Device Sizing for Minimum Energy Subthreshold Circuits. *Proceedings of the International Symposium on Low Power Electronics and Design*, pages 8–13, Oct 2006.

[28] J. Rose and A. Vincentelli. Architecture of Field Programmable Gate Arrays. *Proceedings of the IEEE*, 81(7):1013–1029, 1993.

[29] J. Tschanz, J. Kao, S. Narendra, R. Nair, D. Antoniadis, A. Chandrakasan and V. De. Adaptive Body Bias for Reducing Impacts of Die-to-Die and Within-Die Parameter Variations on Microprocessor Frequency and Leakage. In *Proceedings of the International Solid State Circuits Conference*, pages 1396–1402, February 2002.

[30] N. Lindert, T. Sugii, S. Tang, and C. Hu. Dynamic Threshold Pass-Transistor Logic for Improved Delay at Lower Power Supply Voltages. *IEEE Journal of Solid State Circuits*, 34:85–89, Jan 1999.

[31] M. Berkelaar and J. Jess. Gate Sizing in MOS Digital Circuits with Linear Programming. *Proceedings of the Conference on European Design Automation*, pages 217–221, Jan 1990.

[32] M. Berkelaar, K. Eikland and P. Notebaert. lp_solve 5.5. *Open source (Mixed-Integer) Linear Programming System, GNU LGPL*, 2004.

[33] M. Chen and J. Ho and T. Huang. Dependence of Current Match on Back-Gate Bias in Weakly Inverted MOS Transistors and Its Modeling. *IEEE Journal of Solid State Circuits*, 31(2):259–262, February 1996.

[34] M. Pelgrom and A. Duinmaijer and A. Welbers. Matching Properties of MOS Transistors. *IEEE Journal of Solid State Circuits*, 24(5):1433–1439, October 1989.

[35] J. Montanaro, R. Witek, K. Anne, A. Black, E. Cooper, D. Dobberpuhl, P. Donahue, J. Eno, and D. Kruckemyer. A 160-MHz, 32-b, 0.5-W CMOS RISC Microprocessor. *IEEE Journal of Solid-State Circuits*, 31:1703–1714, 1996.

[36] MOSIS. Global Foundries 8RF-DM. *https://www.mosis.com/vendors/view/global-foundries/8rf-dm-options*.

[37] MOSIS. Kyocera Packages. *https://www.mosis.com/pages/Technical/Packaging/Ceramic/pkg-pga108-connect*.

[38] MOSIS. MOSIS Educational Program. *https://www.mosis.com/you-are/academic-institutions*.

[39] MOSIS. TSMC Processes. *https://www.mosis.com/vendors/view/tsmc/018*.

[40] S. Mutoh, T. Douseki, Y. Matsuya, T. Aoki, S. Shigematsu, and J. Yamada. 1-V Power Supply High-speed Digital Circuit Technology with Multithreshold-Voltage CMOS. *IEEE Journal of Solid State Circuits*, pages 847–854, Aug 1995.

[41] N. Jayakumar and S. Khatri. A Variation-tolerant Sub-threshold Design Approach. *Proceedings of the Design Automation Conference*, pages 716–719, June 2005.

[42] N. Verma and A. Chandrakasan. A 256 kb 65nm Subthreshold SRAM Employing Sense-Amplifier Redundancy. *IEEE Journal of Solid State Circuits*, 43:141–149, 2008.

[43] G. Nabaa, N. Azizi, and F. Najm. An Adaptive FPGA Architecture with Process Variation Compensation and Reduced Leakage. *Proceedings of Design Automation Conference*, pages 624–629, Jul 2006.

[44] Nangate Incorporated. Open Cell Library. *http://www.nangate.com/*.

[45] North Carolina State University. FreePDK. *http://www.eda.ncsu.edu/wiki/FreePDK*.

[46] K. Nowka, G. Carpenter, E. MacDonald, H. Ngo, B. Brock, K. Ishii, T. Nguyen, and J. Burns. A 32-bit PowerPC System-on-a-Chip With Support for Dynamic Voltage Scaling and Dynamic Frequency Scaling. *IEEE Journal of Solid State Circuits*, 37:1441–1447, 2002.

[47] P. Gill, W. Murray and M. Wright. *Numerical Linear Algebra and Optimization*, volume 1. Addison-Wesley Publishing Company, 1991.

[48] R. Sankaranarayanan and M. Guthaus. A Single-Vdd Ultra-Low Energy Sub-threshold FPGA. In *Proceedings of the IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC)*, pages 219–224, 2012.

[49] R. Sankaranarayanan and M. Guthaus. Energy Savings and Performance Improvement in Subthreshold Using Adaptive Body Bias. In *Proceedings of the 27th ACM Great Lakes Symposium on VLSI (to appear)*, 2017.

[50] R. Swanson and J. Meindl. Ion-implanted Complementary MOS Transistors in Low Voltage Circuits. *IEEE Journal of Solid State Circuits*, pages 146–153, Apr 1972.

[51] J. Ryan and B. Calhoun. A Sub-Threshold FPGA with Low-Swing Dual-Vdd Interconnect in 90nm CMOS. *Proceedings of IEEE Custom Integrated Circuits Conference*, pages 1–4, Sep 2010.

[52] S. Hanson, B. Zhai, M. Seok, B. Cline, K. Zhou, M. Singhal, M. Minuth, J. Olson, L. Nazhandali, T. Austin, D. Sylvester and D. Blaauw. Exploring Variability and Performance in a Sub-200mV Processor. *IEEE Journal of Solid State Circuits*, 43:881–891, 2008.

[53] S. Hanson, M. Seok, D. Sylvester and D. Blaauw. Nanometer Device Scaling in Sub-threshold Logic and SRAM. *IEEE Transactions on Electron Devices*, pages 175–185, Jan 2008.

[54] S. Henzler. *Power Management of Digital Circuits in Deep Sub-Micron CMOS Technologies*. Springer Series in Advanced Microelectronics, 2007.

[55] S. Kang and Y. Leblebici. *CMOS Digital Integrated Circuits Analysis and Design*. McGraw Hill, 2003. pages 110-111.

[56] S. Kim and M. Guthaus. Leakage-Aware Redundancy for Reliable Sub-threshold

Memories. In *Proceedings of the Design Automation Conference*, pages 435–440, 2011.

[57] S. Kulkarni, D. Sylvester and D. Blaauw. A Statistical Framework for Post-Silicon Tuning through Body Bias Clustering. In *Proceedings of the International Conference on Computer Aided Design*, pages 39–46, November 2006.

[58] S. Narendra, J. Tschanz, J. Hofsheier, B. Bloechel, S. Vangal, Y. Hoskote, S. Tang, D. Somasekhar, A. Keshavarzi, V. Erraguntla, G. Dermer, N. Borkar, S. Borkar and V. De. Ultra-low Voltage Circuits and Processor in 180nm to 90nm Technologies with a Swapped-body Biasing Technique. In *Proceedings of the International Solid State Circuits Conference*, pages 156–158, February 2004.

[59] S. Srinivasan, A. Gayasen, N. Vijayakrishnan and T. Tuan. Leakage Control in FPGA Routing Fabric. *Procedings of the Asia South Pacific Design Automation Conference*, pages 661–664, Aug 2005.

[60] Synopsys Incorporated. HSPICE. *https://www.synopsys.com/verification/ams-verification/circuit-simulation/hspice.html*.

[61] T. Tuan, A. Rahman, S. Das, S. Trimberger and S. Kao. A 90-nm Low-Power FPGA for Battery-Powered Applications. *IEEE Transactions on Computer Aided Design*, pages 296–300, Feb 2007.

[62] W. Carter, K.Duong, R.H. Freeman, H.C. Hsieh, J.Y. Ja, J.E. Mahoney, L.T. Ngo

and S.L. Sze. A User Programmable Reconfigurable Gate Array. *Proceedings of the IEEE Custom Integrated Circuits Conference*, pages 233–235, May 1986.

[63] Xilinx Corporation. XC3000 Series Field Programmable Gate Arrays. *https://www.xilinx.com/support/documentation/data-sheets/3000.pdf*.

[64] Y. Tsividis and R. Ulmer. A CMOS Voltage Reference. *IEEE Journal of Solid State Circuits*, pages 774–778, Dec 1978.

[65] Y. Ye, F. Liu, M. Chen, S. Nassif and Y. Cao. Statistical Modeling and Simulation of Threshold Variation Under Random Dopant Fluctuations and Line-Edge Roughness. *IEEE Transactions on VLSI*, 19(6):987–996, June 2011.

[66] Y.Lai, C.Kao, T.Chang, and K.Chen. A field programmable gate array with hierarchical interconnection structure. *Proceedings of the International Symposium on Circuits and Systems*, pages 402–405, 1998.