Swarup Bhunia
Saibal Mukhopadhyay
Editors

Low-Power Variation-Tolerant Design in Nanometer Silicon

Springer

Low-Power Variation-Tolerant Design in Nanometer Silicon

Swarup Bhunia · Saibal Mukhopadhyay
Editors

Low-Power Variation-Tolerant Design in Nanometer Silicon

Springer

Editors
Swarup Bhunia
Department of Electrical Engineering
and Computer Science
Case Western Reserve University
Cleveland, OH 44106-7071, USA
skb21@case.edu

Saibal Mukhopadhyay
School of Electrical and
Computer Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0250, USA
saibal@ece.gatech.edu

ISBN 978-1-4419-7417-4 e-ISBN 978-1-4419-7418-1
DOI 10.1007/978-1-4419-7418-1
Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2010938792

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The energy required for running integrated circuits (ICs) is increasing in every new generation of electronic systems. At the same time the manufacturing process used to build these ICs are becoming less deterministic. Hence, low-power design under large parameter variations has emerged as an important challenge in the nanometer regime. The book, for the first time, integrates description of low power and variation issues and provides design solutions to simultaneously achieve low power and robust operation under variations.

Design considerations for low power and robustness with respect to variations typically impose contradictory requirements. Power reduction techniques such as voltage scaling, dual-threshold assignment and gate sizing can have large negative impact on parametric yield under process variations. This book introduces the specific challenges associated with low power and variation-tolerant design in the nanometer technology regime at different levels of design abstraction. It considers both logic and memory design aspects and encompass modeling, analysis as well as design methodology to simultaneously achieve low power and variation tolerance while minimizing design overhead. The issue of device degradation due to aging effects as well as temporal variation in device parameters due to environmental fluctuations are also addressed. Micro-architecture level design modifications, subthreshold design issues, statistical design approaches, design of low-power and robust digital signal processing (DSP) hardware, analog and mixed-signal circuits, reconfigurable computing platforms such as field programmable gate array (FPGA) are covered. The book also discusses emerging challenges at future technology nodes as well as methods industrial and academic researchers and engineers are developing to design low-power ICs under increasing parameter variations.

Why a New Book

Designing low power but reliable ICs using inherently unreliable process is a key challenge in current generation IC design. The problem is escalating in every new generation of ICs. The students, industrial/academic researchers, and practicing engineers need to have basic understanding of this problem and knowledge of existing and emerging methods to enable designing ICs using nanoscale processes. Currently, there exists no book that specifically addresses this problem. Considering the importance of the topic, the book compiles outstanding contributions from prominent personalities in this area covering different aspects of low power and variation-tolerant design.

Unique Features

- Introduction to the most important challenges in the area of low power and variation-tolerant IC design in nanoscale technologies.
- A holistic view of the solution approaches at different levels of design abstraction covering device, circuit, architecture, and system.
- Comprehensive coverage on both logic and memory circuits.
- Comprehensive coverage of system-level design – micro-architecture, DSP, mixed-signal, and FPGA.
- Modeling and analysis of different variation effects (die-to-die and within-die, process and temporal) on low power designs.
- Description of ultralow power and robust subthreshold design aspects.
- Description of technology scaling trends, emerging challenges, and future research directions.

Organization and Overview of the Content

This book is organized in four integrated parts, each covering a major aspect of low power and variation tolerance. The content of each part as well as each chapter in the part is designed to provide necessary background to the readers new in the field as well as detailed discussions on state-of-the-art techniques for the experts in the topic. Part I is dedicated to provide the reader a comprehensive background of the two complementary problems – power dissipation and parameter variations. The backgrounds are supplemented with description of existing techniques for power estimation and optimization as well as variation modeling and characterization. The rest of the book aims at establishing the principle that the challenge of low power under process variations needs to be addressed at all levels of design abstraction, i.e., from circuits to logic to micro-architecture to system. Part II focuses on circuit and logic level techniques including computer-aided design methods; Part III discusses system-level methods for low power under process variations for microprocessors, digital signal processors, and analog/RF/mixed-signal systems. Part IV discusses the power and variation challenges for reconfigurable computing platforms and unique design methods to address them, particularly focusing on Field-Programmable-Gate-Arrays (FPGA). We will now provide a brief overview of each part of this book and the corresponding chapters.

Part I: Physics of Power Dissipations and Parameter Variations

Part I provides a comprehensive background of the two complementary problems – power dissipation and variations. This part is organized in two chapters – one dedicated to basics of power dissipation and second dedicated to the basics of process variations. Chapter 1 discusses the different sources of variation in CMOS technologies and presents an overview of how to characterize these variations. The chapter covers various sources of process variation during manufacturing; sources of environmental variations including temperature and voltage fluctuations; and finally, sources of temporal variations. Different state-of-the-art characterization circuits and sensors employed in modern integrated circuits to understand the extent and impact of variations are presented. An overview of the power dissipation issues of the state-of-the-art CMOS technologies is presented in Chapter 2. This chapter investigates the sources of IC power dissipation and techniques for power analysis in nanometer CMOS circuits. Finally, different power optimization techniques from circuit and physical design to system synthesis are discussed.

Part II: Circuit-Level Design Solutions

Part II discusses the circuit-level challenges and design methodologies for low-power and process variation tolerance. This part is organized in four chapters – logic circuits, statistical design, memory circuits, and ultra-low-voltage design.

The process and temporal variations prevent a logic block from meeting timing and power criteria. The net effect is degradation in the parametric yield. The low-power design methods like voltage scaling, dual threshold assignment further magnify the effect of variations. A trivial solution to the problem of variation is over-designing the logic circuit, for example, using transistor upsizing or supply voltage boosting. But over-design comes at a serious power cost. Researchers at industry and academia have been exploring methods that would allow variation tolerance at no or minimum power overhead. Considering the complexity of the current day integrated circuits, these methods span the domains of circuit design as well as computer-aided design (CAD) methodologies. Chapter 3 reviews circuit-level methods to address this goal. The chapter first analyzes the effect of process variations and time-dependent degradation mechanisms. The analysis is followed by discussion of two design approaches to address these problems: variation tolerant circuits and adaptive circuits that tune themselves to operate correctly under variations. Chapter 4 of this part focuses on the statistical design paradigm, as opposed to the traditional corner-based methodology, to address the variation problem. The techniques for pre-silicon statistical timing and power analysis are presented to determine the performance spread of large logic blocks due to variations. Next, the pre-silicon statistical optimization techniques are discussed to improve the parametric yield. Finally, the chapter discusses how a set of compact sensors may be used to predict the delay of a manufactured part and drive adaptive post-silicon tuning.

The die-to-die or within-die spatial variations in process parameters are the primary cause of concerns in logic circuits which lead to delay or leakage variations. On the other hand, the memory circuits are more susceptible to local random variations that results in different types of parametric failures. Chapter 5 discusses the impact of process variations on reliability of Static Random Access Memory (SRAM). The chapter provides an overview of the mechanisms of SRAM parametric failures and estimation methods for failure probabilities and parametric yield of SRAM. This analysis is followed by design approaches for SRAM yield enhancement including cell sizing, redundancy, dynamic circuit techniques, post-silicon adaptive repair techniques, and variation-tolerant SRAM peripherals. Finally, a discussion on adaptive low power and variation-tolerant SRAM design for multimedia applications is provided.

This part concludes with the discussion of a relatively new concept of digital design – the subthreshold design – where operating voltage of the circuit is lower than the threshold voltage of the transistor. The applications of the subthreshold design in ultra-low-power electronics are being actively investigated. Chapter 6 provides a brief overview of the principles and challenges subthreshold digital design. Design principles at all levels of hierarchy, namely, devices, circuits, and architecture, need to be evaluated for maximum power gains. Brief description of SRAM design techniques as well as alternative architectures for lower power in subthreshold design has also been discussed.

Part III: System-Level Design Solutions

Part III focuses on system-level design challenges and design methodologies for low-power and process-variation tolerance. The three chapters cover three different types of mainstream electronic systems: microprocessor, application specific integrated circuits such as digital signal processors, and analog/mixed Signal/RF circuits.

Chapter 7 discusses micro-architectural techniques to tolerate variations in microprocessors and design variation-tolerant low-power systems. The chapter discusses how different variation sources impact the timing uncertainty at the system level. Designers typically account for parameter variations by inserting conservative margins that guard against worst-case variation characteristics to guarantee functional correctness of the system under all operating conditions. But such conservative approaches lead to performance degradation. This chapter presents alternative error-tolerant schemes to deal with these different sources of parameter variations. Error-tolerant schemes aim to run with nominal timing-margins but without comprising robustness and correctness.

Chapter 8 presents an overview of low power and variation tolerant challenges in digital signal processing (DSP) systems. The chapter discusses how to exploit properties of DSP algorithms and combine circuit and architecture-level techniques in order to provide intelligent trade-offs between circuit-level metrics such as power and performance with system-level metrics such as quality-of-results and tolerance to variations (yield). The techniques presented in this chapter target various types of designs that include logic and memory architectures and complete DSP systems.

Chapter 9 discusses challenges in analog/mixed Signal/RF systems and complements the methodologies throughout the book for digital circuits and systems. Low-Power and variation-tolerant analog/mixed Signal/RF circuits require very different set of design techniques. The design techniques range from individual circuit components to system-level methods. This chapter highlights the component level as well as system-level design challenges in a low-power, process-tolerant mixed Signal/RF system using a wireless transceiver as a demonstration vehicle. At the component level, variation-tolerance for critical blocks such as analog-to-digital conversion, low-noise amplifiers are discussed. At the system level, the chapter focuses on how to achieve low power and end-to-end quality-of-service in RF transceiver under both manufacturing and environmental variations.

Part IV: Low-Power and Robust Reconfigurable Computing

Part IV concentrates on low power and variation tolerance in reconfigurable computing platforms. FPGAs are reconfigurable devices that can be programmed after fabrication to implement any digital logic. Compared to application-specific integrated circuits or ASIC, FPGA provides flexibility, in-field reconfigurability, and potential cost advantage but at the expense of area, performance, and perhaps most importantly power. FPGAs are normally less power efficient than ASICs but significant research efforts devoted to improving power efficiency of FPGA can potentially narrow or even close this gap. Chapter 10 surveys the techniques and progress made to improve FPGA power efficiency. Parameter variation and component aging are becoming a significant problem for all digital circuits including FPGAs. These effects degrade performance, increase power dissipation, and cause permanent faults at manufacturing time and during the lifetime of an FPGA. Chapter 11 examines the impact of variation and device wear-out on FPGAs. The chapter discusses different techniques that can be used to tolerate variations in FPGA with specific attention to how the reconfigurable nature of the FPGAs can lead to innovative solutions to variation and wear-out management. Chapter 12 discusses design methods that can provide both low power and variation tolerance. This chapter discusses the opportunities of using post-fabrication reconfiguration in FPGA to tolerate variation effects without expensive static margins. The post-fabrication reconfigurability can be used to deploy devices based on their fabricated or aged characteristics to place the high-speed/leaky devices on critical paths and slower/less-leaky devices on non-critical paths. The component-specific mapping methodologies that can achieve the above objectives are discussed in the chapter.

We believe the target readership consisting of students, researchers, and practitioners will like the content and be greatly benefited from it. We also believe that the content will remain highly relevant to emerging technologies – both silicon and non-silicon – since power and variability continue to be major design issues in the nanometer regime. We are very grateful to all the outstanding contributors for providing the high-quality chapters for the book. We highly appreciate their whole-hearted cooperation throughout this project. We also acknowledge the help and support from the students, in particular from Somnath Paul, Seetharam Narasimhan, and Anandaroop Ghosh of Nanoscale Research Lab at Case Western Reserve University as well as students from the GREEN Lab, at Georgia Institute of Technology. We also remain sincerely thankful to Springer, USA and all their publication stuffs.

Cleveland, Ohio Swarup Bhunia
Atlanta, Georgia Saibal Mukhopadhyay

Contents

Part I Physics of Power Dissipations and Parameter Variations

Part II Circuit-Level Design Solutions

Part III System-Level Design Solutions

Contributors

Aditya Bansal IBM T.J. Watson Research Center, Yorktown Heights, NY, USA, bansal@us.ibm.com

Pradip Bose IBM T.J. Watson Research Center, Yorktown Heights, NY, USA, pbose@us.ibm.com

Abhijit Chatterjee Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA, chat@ece.gatech.edu

André DeHon Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA, USA, andre@seas.upenn.edu

Swaroop Ghosh Intel Corporation, Portland, OR, USA, swaroopad.ghosh@intel.com

Benjamin Gojman Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA, bgojman@seas.upenn.edu

Meeta S. Gupta IBM T.J. Watson Research Center, Yorktown Heights, NY, USA, msgupta@us.ibm.com

Georgios Karakonstantis Department of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA, gkarakon@purdue.edu

Hamid Mahmoodi Department of Electrical and Computer Engineering, San Francisco State University, San Francisco, CA, USA, mahmoodi@sfsu.edu

Nikil Mehta Department of Computer Science, California Institute of Technology, Pasadena, CA, USA, nikil@caltech.edu

Vishwanath Natarajan Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA, vishwa@ece.gatech.edu

Bipul C. Paul Global Foundries, Fishkill, NY, USA, bipul.paul@globalfoundries.com

Rahul M. Rao IBM T.J. Watson Research Center, Austin, TX, USA, raorahul@us.ibm.com

Kaushik Roy Department of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA, kaushik@ecn.purdue.edu

Arijit Raychowdhury Intel Corporation, Portland, OR, USA, arijit.raychowdhury@intel.com

Raphael Rubin Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA, rafi@seas.upenn.edu

Sachin S. Sapatnekar Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis, MN, USA, sachin@umn.edu

Shreyas Sen Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA, shreyas.sen@gatech.edu

Li Shang Department of Electrical, Computer, and Energy Engineering, University of Colorado, Boulder, CO, USA, li.shang@colorado.edu

James Williamson Department of Electrical, Computer, and Energy Engineering, University of Colorado, Boulder, CO, USA, james.williamson@colorado.edu

Wei Zhang School of Computer Engineering, Nanyang Technological University, Singapore, zhangwei@ntu.edu.sg

About the Editors

Swarup Bhunia received his B.E. (Hons.) from Jadavpur University, Kolkata, and the M.Tech. degree from the Indian Institute of Technology (IIT), Kharagpur. He received his Ph.D. from Purdue University, USA, in 2005. Currently, Dr. Bhunia is an assistant professor of electrical engineering and computer science at Case Western Reserve University, Cleveland, USA. He has published over 100 articles in peer-reviewed journals and premier conferences in the area of VLSI design, CAD, and test techniques. His research interests include low power and robust design, hardware security and protection, adaptive nanocomputing, and novel test methodologies. He has worked in the semiconductor industry on RTL synthesis, verification, and low power design for about 3 years. Dr. Bhunia received Semiconductor Research Corporation (SRC) technical excellence award (2005), SRC Inventor Recognition Award (2009), best paper award in International Conference on Computer Design (ICCD 2004), best paper award in Latin American Test Workshop (LATW 2003), and best paper several best paper nominations. He has served as a guest editor of IEEE Design and Test of Computers (2010), and Journal of Low Power Electronics (JOLPE). He has also served in the technical program committee of many IEEE and ACM conferences. He is an Associate Editor of ACM Journal on Emerging Technologies (JETC) and a senior member of IEEE.

Saibal Mukhopadhyay received the B.E. degree in electronics and telecommunication engineering from Jadavpur University, Calcutta, India, in 2000 and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, in 2006. He was with the IBM T.J. Watson Research Center, Yorktown Heights, NY, as a research staff member. Since September 2007, he is with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, as an assistant professor. His research interests include analysis and design of low power and robust circuits in nanometer technologies. Dr. Mukhopadhyay was the recipient of the IBM Faculty Partnership Award in 2009 and 2010, SRC Inventor Recognition Award in 2008, IBM Ph.D. Fellowship Award in 2004, the SRC Technical Excellence Award in 2005, the Best in Session Award at the 2005 SRC TECNCON, and the Best Paper Awards at the 2003 IEEE Nano and the 2004 International Conference on Computer Design. He is a coauthor of more than 100 papers in refereed journals and conference proceedings, and co-inventor of three US Patent.

Part I
Physics of Power Dissipations and Parameter Variations

Chapter 1
Variations: Sources and Characterization

Aditya Bansal and Rahul M. Rao

Abstract This chapter discusses the different sources of variation which can deviate a circuit's characteristics from its intended behavior. First we discuss the sources of process variation during manufacturing, followed by environmental variations during usage. Environmental variations include temperature, voltage fluctuations, and temporal variations. Finally, we discuss the state of art characterization circuits (or sensors) employed to understand the extent and impact of variations.

1.1 Introduction

Traditionally, performance benchmarking and power estimation are based on the premise that the electrical characteristics and operating conditions of every device in the design matches the model specifications. However, with continued scaling of device dimensions to sub-100-nm regime, it has become nearly impossible to maintain the same level of manufacturing control and uniformity. This can cause devices to behave differently from model characteristics. Further, devices that were intended to be identical could differ vastly in their electrical characteristics which can lead to functional failures. Environmental factors such as supply voltage and temperature experienced by devices in different parts of the chip (or on different chips) also vary due to different levels of device densities, switching activities, and noise integrity of the various blocks. The voltage and temperature stress experienced by the devices with continued usage degrades their electrical characteristics, thereby increasing the mismatch between the idealistic models and the actual device parameters. All these factors can combine to make the actual design considerably different from the intended design. The performance of the design can thus vary and be lower than the intended one. Similarly, due to the exponential dependence between process/device parameters and transistor leakage, the chip power can also vary and be significantly higher than the nominal values. This translates into a reduced

A. Bansal (✉)
IBM T.J. Watson Research, Yorktown Heights, NY, USA
e-mail: bansal@us.ibm.com

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_1,

parametric yield, which is the number of manufactured chips that satisfy the required performance, power, and reliability specifications, and hence limits the number of shippable products. With the initial design cost being the same, the cost per good chip increases which has a direct impact on the bottom line dollar value of the design.

The impact of these variations on the parametric yield is generally reduced by guard-banding to provide sufficient tolerance margins during design, which is equivalent to designing at a non-optimal power-performance point. Designs are rated to run at lower than nominal frequencies to guarantee functional correctness in the presence of variations. With increasing variation, the required guard-banding also increases, especially if worst-case conditions are considered. Understanding the various sources of variation and their impact on circuits can help in determining a realistic amount of design margining required, and also provide insights to mitigate the effects of variation. On-chip characterization circuits can help in determining the extent of variation, isolating various effects and their dependence and providing feedback to the manufacturing team. They can also be used in the field to monitor the chip periodically, detect possible failure conditions, and adjust global parameters that can help compensate against these effects.

Variations can be broadly classified into process, environmental, and temporal variations. *Process variations* are a consequence of imperfect control over the fabrication process. *Environmental variations* arise during the operation of the circuit due to changes in the operating conditions such as temperature and supply voltage. *Temporal variations* refer to the change in the device characteristics over time. The rest of the chapter will provide an insight into the sources of variation and circuits commonly used to characterize these effects.

1.2 Process Variations

Process variations occur due to lack of perfect control over the fabrication process. These variations always existed since the invention of transistor; however, they are increasingly gaining importance with the scaling of FET dimensions – especially in sub-100-nm technology nodes. In today's VLSI circuits, no two FETs on a chip are exactly same at the atomic level – even neighboring devices can be significantly different. As an illustration, Fig. 1.1 shows the correlation in frequency among four identically designed performance screen ring-oscillators (PSROs) on a chip. It can be seen that the frequencies of neighboring PSROs are correlated to each other; however, not exactly matched though identically designed. This correlation depends on the technology, layout, and the fabrication imperfections.

1.2.1 Sources of Variations

Process variations can be categorized as systematic or statistical in nature. Systematic variations repeat from chip-to-chip and cause fixed offset from design target in FETs. They are typically caused by errors in mask build, optical proximity

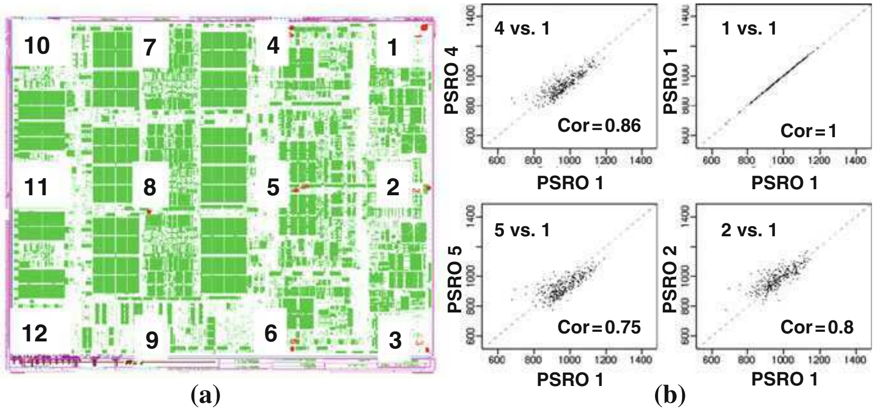


Fig. 1.1 (**a**) Chip (~250 mm^2) map showing the location of 12 identical ring-oscillators. (**b**) Correlation corner (averaged over 300 chips) for 4 ring-oscillators. (Courtesy: Manjul Bhushan, IBM)

correction (OPC), layout, tool optics, etc. Statistical variations make the two chips or two devices different. Range of statistical variations can vary from ~300 mm to less than 10 nm. Figure 1.2 summarizes the range of key process variations. Wafer-scale variations (may be due to chuck holding the wafer during process steps) cause variations in wafers but all the chips on a particular wafer see the same effect. Similarly, chip-level variations cause two chips to differ from each other; however, all the FETs

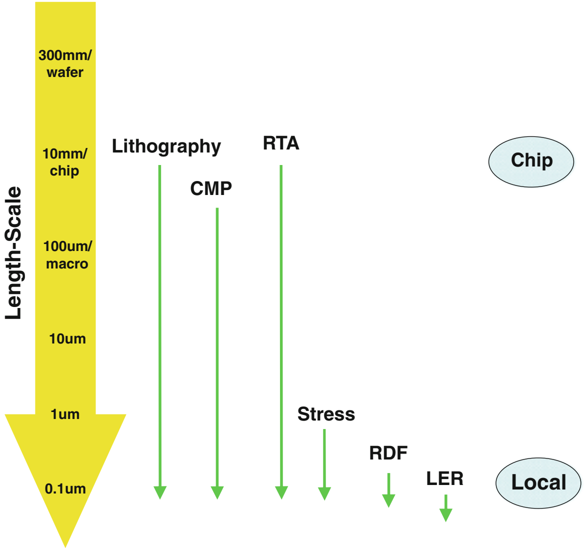


Fig. 1.2 Variation range of various mechanisms

on one chip see the same affect. Medium range variations (10–100 μm) cause variations from one circuit block to another circuit block. For example, one section of a chip can become very slow whereas another section can be very leaky. Small range variations (< 1 μm) are typically intrinsic and caused by random dopant fluctuations (RDF) and line-edge roughness (LER). These variations cause two neighboring FETs to have different electrical characteristics.

In this section, we will focus on the sources of statistical process variations and their impact on electrical characteristics. We broadly classify sources of variations into two categories – tool-based and intrinsic. Tool-based variations are medium to long-range variations whereas intrinsic variations are small variations.

1.2.2 Fabrication Tool-Induced Variations

1.2.2.1 Lithographic Variations

Lithographic variations occur due to the imperfect tools and depend on the physical design. Figure 1.3 shows a simplified view of a lithographic tool used to pattern the FET gate, active diffusion region, contacts, vias, and interconnects. First masks are

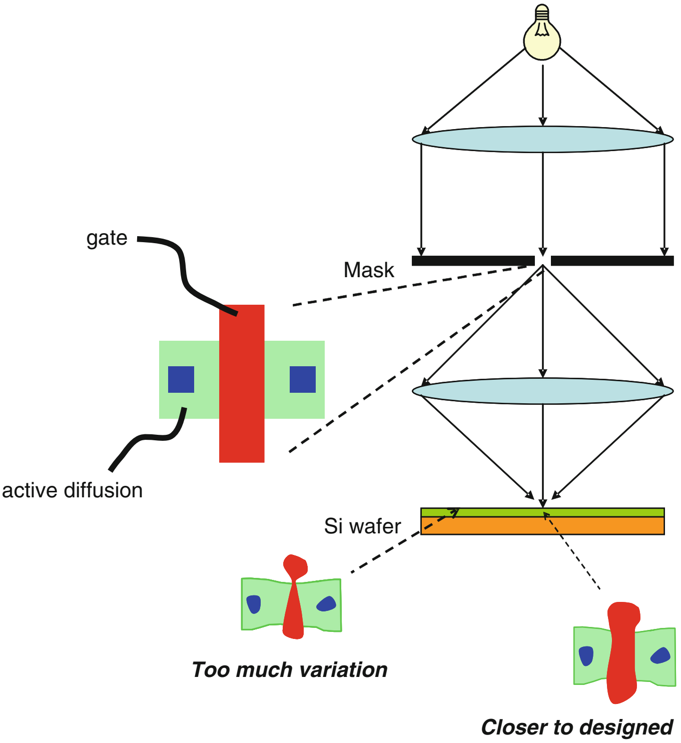


Fig. 1.3 Schematic of a lithographic tool

generated to represent the designed patterns in each fabrication layer. VLSI circuits are highly complex and the mask formation is largely an automated process. Today, sub-50-nm dimensions are printed on a wafer using a light source with wavelength 193 nm. Several advanced mathematical algorithms and techniques, such as optical proximity correction (OPC) [1], (RET), etc., are used to reliably print a pattern. During the mask generation process, some errors can occur resulting in ill-formation of the pattern. These are called *mask errors*. These errors are random in nature and may or may not exist. For example, mask-error can occur for a specific shape of poly silicon resulting in narrow gate of a FET. After a mask is generated, it is used to print a pattern on the wafer. For sub-50-nm dimensions on a wafer, lenses have to be perfectly aligned and focused. However, it is not possible to perfectly control the focus on the whole wafer resulting in reduced or increased dimensions. For example, *focus variation* in a gate formation can either narrow the gate length or increase it. Further, the dosage of light source is very critical and it is not uniform across the full wafer resulting in the variation in critical dimension (CD) due to *dose variations*. These lithographic variations are typically of the order of tens of microns. These statistical variations can be assumed to have Gaussian distribution. Typically we look at the variation in CD for ±3σ variations in lithography.

For circuit analysis, an effective gate length needs to be computed for each FET considering process fluctuations. There are several methods devised to account for non-rectangularity in FETs and include them in circuit simulations. Heng et al. [2] proposed one such method which can be used to estimate lithography-aware equivalent gate length and width. This method was used in [3] to design a virtual fabrication environment to maximize circuit yield. Figure 1.4 shows the percentage deviation of gate width and length from the design value in a 45-nm technology as computed using this method. In this example, designed gate length is 40 nm and gate width is 200 nm.

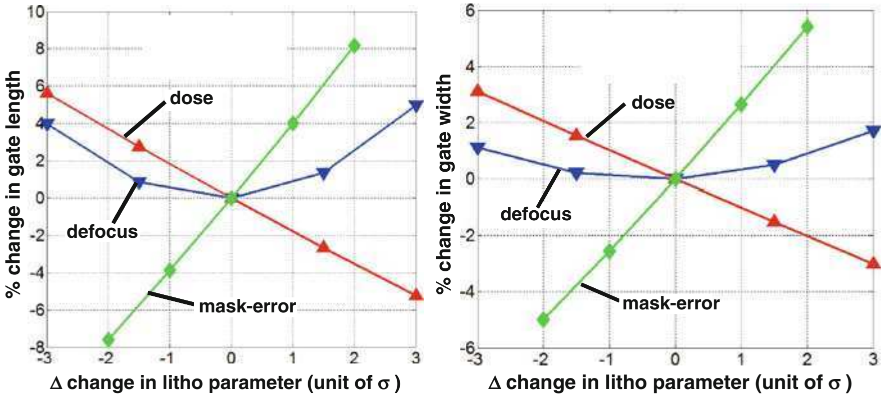


Fig. 1.4 Variation in a FET's length and width due to variation in lithography parameters – defocus, dose, and mask errors. Percentage change in gate length/width, from designed, are computed using the method proposed by [2]

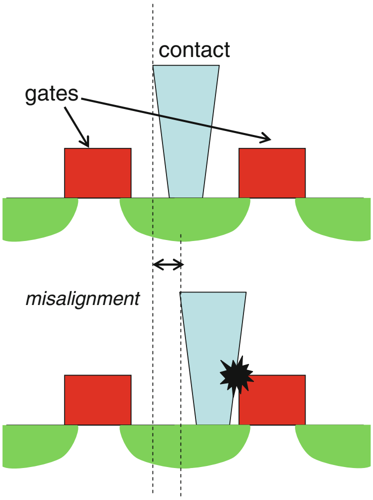


Fig. 1.5 Implication of contact and gate misalignment

Another source of variation can arise from wafer misalignment from one process to another. One critical example of misalignment-induced variation or defect is the distance between a gate and a contact (Fig. 1.5). For example, gate pitch of 130 nm with gate length of 40 nm leaves 90-nm distance between the two neighboring gate edges for contact formation. For a contact size of 40 nm × 40 nm, the closest distance between the gate edge and the contact edge is 25 nm. The size of the contact cannot be reduced to keep the low resistance while gate pitch cannot be increased to maintain density. Hence, increase in wafer misalignment will increase the gate to contact capacitance eventually leading to catastrophic failure – short between gate and contact.

1.2.2.2 Chemical Mechanical Polishing (CMP)

CMP is used for planarizing the metal interconnect layer or inter-layer dielectric (ILD) between adjacent metal layers due to copper damascene process. Figure 1.6 shows the variation in interconnect thickness in interconnect layer post-CMP. Metal CMP increases resistance due to loss of metal and decreases intra-layer coupling among adjacent wires. Increased resistance is somewhat compensated by decrease in capacitance for circuit delay. ILD CMP introduces change in inter-wire capacitances. Hence, CMP brings variation in the delay of interconnects resulting in non-deterministic circuit performance from chip-to-chip and within chip.

Other fabrication steps which can induce variation are rapid thermal annealing (RTA) [4] and stress liner effect [5]. The key FET parameters prone to these variations are length and width.

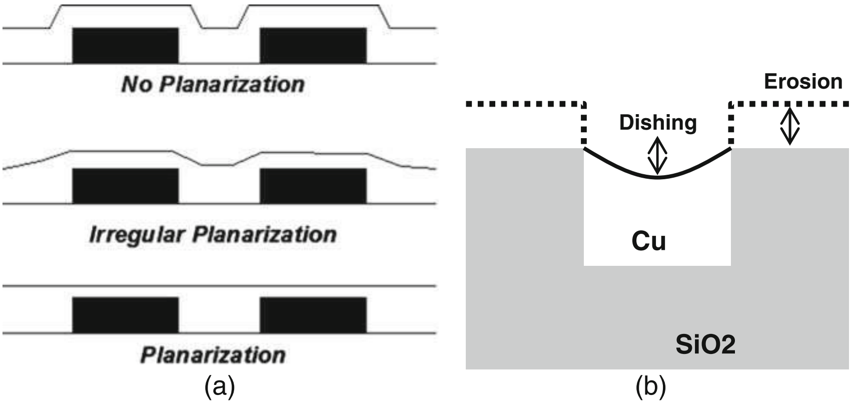


Fig. 1.6 (**a**) Planarization by CMP and (**b**) dishing and erosion causing variation in interconnect thickness

1.2.2.3 Variation in Contact Resistance

Contact resistance is becoming critical for circuit analysis because of increasing resistance with contact dimension scaling. Figure 1.7 shows the distribution of contact resistance as analyzed by Sleight et al. [6]. They proposed an analytical cone model to compute the resistance under variation in bottom diameter, height and angle of a contact. Variation in contact resistance between two FETs can result in a significant difference in voltages seen by the terminals of these FETs. As an example, let us consider two 1-µm-wide FETs (located spatially far off) such that their source contact resistances differ by 100 Ω. For 1 mA of drain-to-source current,

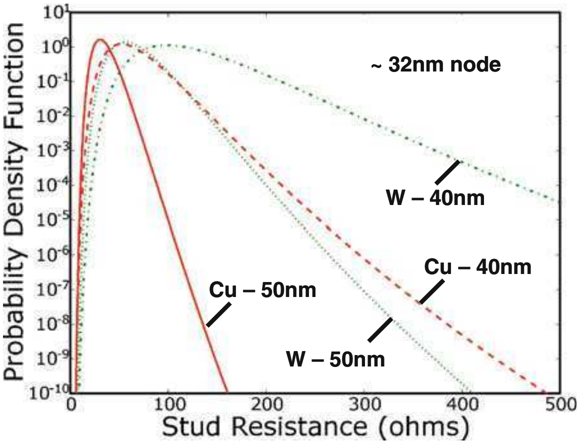


Fig. 1.7 Distribution of contact resistance due to variation in – angle, bottom diameter, and height for a designed contact diameter. Shown are the two types of contacts – Tungsten (W) and Copper (Cu). W contacts are easy to manufacture. *Source*: [6]

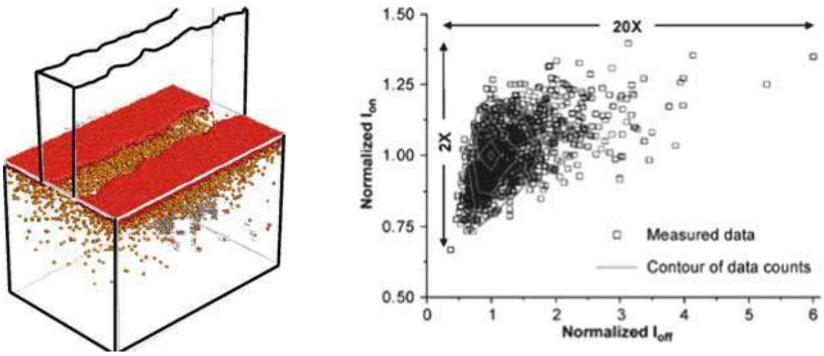
the difference in voltage drops at the source terminals will be 100 mV between the two FETs. This will bring a difference of 100 mV between effective gate-to-source voltages, directly translating to the V_T variation of 100 mV between two FETs. An analytical cone model [6] can be used to compute the contact resistance.

1.2.3 Intrinsic Variations

Beyond variations due to imperfect fabrication tools, some variation sources are intrinsic to a technology. Two key sources of variation which are truly random in nature are random dopant fluctuation (RDF) and line-edge roughness (LER).

1.2.3.1 Random Dopant Fluctuations (RDF)

FETs are doped with impure atoms to control the transistor's electrical properties, primarily short-channel effect and threshold-voltage. With technology scaling, the device volume is reducing thereby reducing the number of dopant atoms in a FET's channel. In sub-100-nm transistors, the position and the number of dopant atoms are critical. When a FET is doped with impurity, the number of dopant atoms and their placements cannot be controlled to the last atom. Moreover, dopant atoms diffuse in the Si channel and randomly get placed as illustrated in Fig. 1.8a. This random number and placement of dopants bring uncertainty in threshold voltage which is called RDF-induced V_T variation. This variation is a grave concern in very small FETs because it is very unlikely to have two neighboring FETs with same number and placement of dopants. The statistical distribution of V_T due to RDF has been found to follow normal distribution. The standard deviation (σV_T) of these distribution scales with FET area following inverse square root law. For example, when a FET scales from L_1, W_1 to L_2, W_2, the standard deviation changes to



© 2003 IEEE

(a)

© 2009 IEEE

(b)

Fig. 1.8 (**a**) RDF [7] and (**b**) distribution of Ion and Ioff in a 65 nm technology [8]

$$\sigma_{V_\mathrm{T}}|_{L_2,W_2} = \sqrt{L_1W_1 \Big/ L_2W_2}\ \sigma_{V_\mathrm{T}}|_{L_1,W_1} \tag{1.1}$$

implying that if a FET's area is reduced by half, the σV_T increases by a factor of 1.4 times. Figure 1.8b shows variation in on and off currents due to RDF-induced V_T fluctuation [8]. As can be understood, large area devices have tighter (lower σV_T) normal distribution of V_T. σV_T is the same for two FETs with same device widths irrespective of the number of fingers. The distribution of V_T due to RDF can be obtained by atomistic Monte-Carlo simulations [9, 10].

1.2.4 Line-Edge Roughness (LER)

The gate edge of a MOSFET is not smooth; instead it is rough (as shown in Fig. 1.9a) primarily due to material properties, imperfect lithography and etching tools. Conventionally, polycrystalline silicon (polysilicon) is used to make gate. Polysilicon material has varying grain size, typically in the range of 5–10 nm. Current fabrication tools are unable to smooth this edge resulting in LER. Similar roughness is expected in metal gates due to the properties of metal used and fabrication steps. As shown in Fig. 1.9a, in FETs with gate length 90 nm or higher, the intrinsic fluctuation in gate length due to LER is less prominent compared to 22 nm technologies. In sub-50 nm technologies, the LER can be critical because FETs are typically designed at the knee of V_T roll-off curve. Figure 1.9b shows that tighter control over the gate length is required in 20 nm long FETs compared to in 65-nm long FETs. The impact of LER can be analyzed by doing 3D TCAD simulations [11]. The distribution of LER-induced V_T variation is Gaussian and scales with gate width following inverse square root law.

$$\sigma_{V_\mathrm{T}}|_{W_2} = \sqrt{W_1/W_2}\ \sigma_{V_\mathrm{T}}|_{W_1} \tag{1.2}$$

implying that the impact of LER will be less in wider transistors.

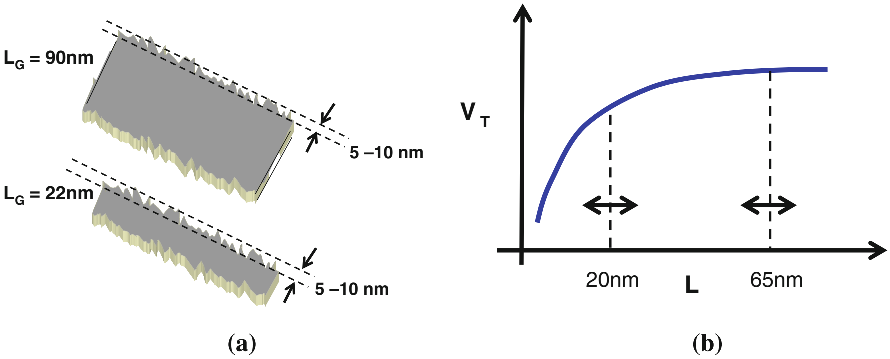


Fig. 1.9 (**a**) Line Edge Roughness (**b**) V_T roll-off curve

1.3 Temperature and Supply Voltage Variations

The electrical behavior of devices in a circuit is strongly dictated by the operating conditions it experiences. The current through a device is determined by the voltage at its terminals, and hence influenced by the operational supply voltage. The threshold voltage (V_{TH}), and carrier mobility (μ) are dependent on the operating temperature. In addition to process variations described in the previous section, any change in these environmental conditions also causes a deviation of the power and performance characteristics of devices from their ideal behaviors.

1.3.1 Temperature Variations

The spatial temperature variation across the die or the temporal difference at the same location on the die depends on several factors. These include:

1.3.1.1 Power Characteristics of the Neighboring Blocks

At a circuit level, the switching activity and the load characteristics of the blocks in the regions around that location determine the power consumption and hence the heat generated near that location. Highly capacitive blocks have a greater current demand and experience a relatively higher temperature in comparison to low-capacitive blocks. Further, blocks with a high-switching activity (e.g., clock-sector buffers) also generate a larger amount of heat in comparison to the blocks with minimal switching activity (e.g., inside a SRAM array or a power-gated block).

1.3.1.2 Thermal Characteristics of Materials

For a given power density, the silicon temperature is a function of the thermal conductivity of the materials used. In a bulk CMOS technology, the heat generated spreads through the silicon substrate as well as the wires. However, in a silicon-on-insulator (SOI) technology the poor thermal conductivity of the buried oxide causes most of the heat to be carried away primarily along the wires, which causes the temperature to increase at a faster rate. This results in a greater variation in the temperature gradient between the power-hungry hotter regions and the low-power cooler regions of the chip.

1.3.1.3 Cooling and Packaging Efficiency

The cooling efficiency of the system also determines the spatial temperature variation. In a conventional design, the rate of temperature increase is determined by the package and heat sink design and the cooling mechanism, with a water-cooled system seeing a better thermal profile than an air-cooled system. However, the temperature variations are likely to be exacerbated in a three-dimensional (3-D) stack technology, where in the dies further away from the heat sink experience a much reduced cooling efficiency and hence experience a large temperature gradient between the hot-regions and the cold-regions of the chip.

1.3.1.4 Switching Activity and Workload Management

Another factor that determines the spatial temperature variation is the actual workload (or application) being run on the system. Certain blocks are continuously exercised in some workloads. In the current era of multi-core processors, it is likely that some of the cores are inactive and hence cold for a greater percentage of time. Depending on the ability of the system to dynamically manage the workload by periodically assigning tasks to different cores, or moving tasks from one core to another, the temperature difference between the cores will vary.

As a result, the temperature on different parts of the die can be vastly different a given point of time. As an example, Fig. 1.10 shows the temperature profile on an IBM Power 4, with a nearly 20°C difference between the hottest and the coolest portions of the chip. With the power densities of current generation designs being much higher, this spatial temperature variation is likely to be even higher, causing a significant difference in the performances of devices across the chip.

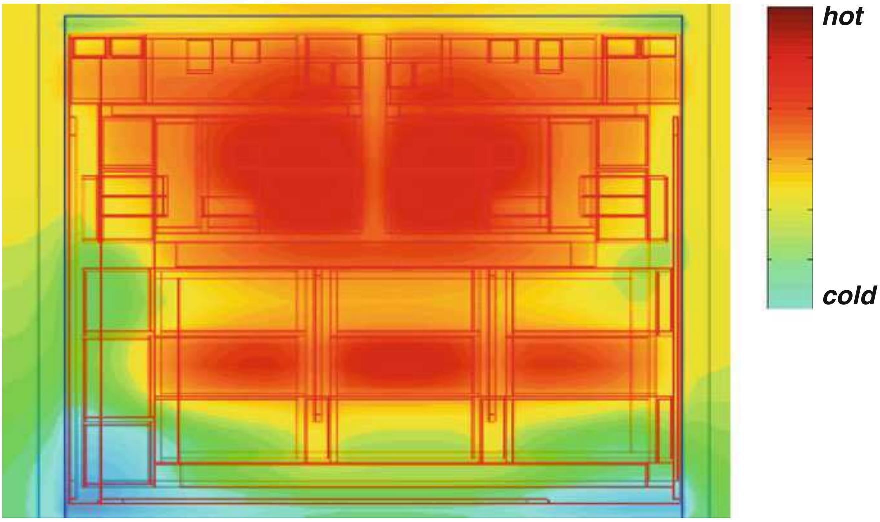


Fig. 1.10 Temperature difference across regions of IBM Power 4

1.3.2 Supply Voltage Variations

The supply voltage at any location on a die deviates from the nominal design value depending on the characteristics of the power grid network and the current requirement (both steady state and transient) of the design. Increased device densities and operating frequency of highly pipelined systems have increased the current requirement of current designs. This, along with worsening wire properties, has made the problem of supply voltage variations more significant.

1.3.2.1 Ldi/dt Effect

Various blocks in a design usually share a single power grid. Any sudden increase in the switching activity of a particular block requires additional current to be supplied to that area of the chip. The parasitic inductance in the power grid network and the package causes a *di/dt* drop and hence a transient voltage drop on the supply lines. With aggressive power management techniques, such as clock-gating and power-gating becoming prevalent in designs, the likelihood of such transient current requirements and its magnitude has increased. Further, with multi-core processors under a common power grid becoming the norm, dynamic changes in the current requirement of any one of the cores can create supply voltage variations on all of the others. This can be seen from Fig. 1.11 which shows the droop in the supply voltage for Core 1 and Core 2 in the scenario that Core 2 turns on a few ns after Core 1. In both the scenarios, there is significant supply voltage droop. However, Core 2 has a much worse voltage droop than Core 1, since the voltage droop from Core 1 is coupled to Core 2 just as it turns on, creating an increased supply voltage noise effect. Such peak–voltage droop scenarios are very difficult to predict, model, or avoid and hence require sufficient guard-banding in the design. The impact of such scenarios can be reduced by proper and sufficient use of on-chip decoupling capacitors and regulators.

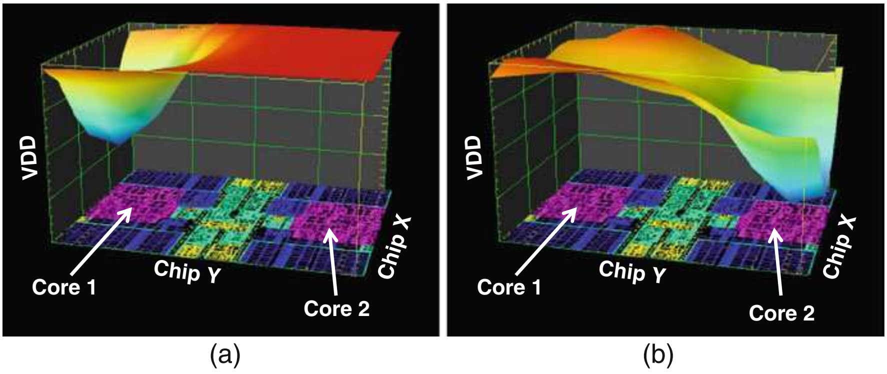


Fig. 1.11 Power supply droop in a multi-core processor for Core 1 and Core 2. (**a**) Voltage droop in Core 1 (left) as it turns on, followed by (**b**) voltage droop in Core 2 (right) as it turns on a few nano-seconds after Core 1 [12]

1.3.2.2 IR Drop

There is also a voltage drop across the power grid due to the resistance of the power supply lines, which is a function of the steady state current carried through the network. This is referred to as the IR drop and is equivalent to the steady state current drawn from the power supply times the resistance of the power grid. Continued scaling of wire dimensions to match device scaling has increased the resistance per unit length of wire. In addition, growing die sizes with larger transistor counts (especially on-die caches) has increased the steady state (leakage) currents exacerbating the IR drop problem. The resistivity of the wires also increases with temperature, which further worsens the IR drop.

1.3.3 Impact on Circuit Performance

Let us now briefly look into the sensitivity of device characteristics and circuit metrics to these environmental variations.

The mobility of charge carriers degrade with temperature due to the increased scattering in the channel. On the other hand, the semiconductor band gap is known to decrease with temperature. Due to the combination of these contradicting effects, the saturation current (and hence the delay of the worst path in the design) can either improve or worsen. However, in the weak inversion region, the effect of decreased band gap and its exponential influence on the sub-threshold leakage current results in a substantial increase in leakage currents for a small change in temperature. Further, all junction leakages also increase with temperature. Hence, the leakage power significantly increases with temperature making it a critical challenge especially for low-power designs in the hand-held and portable market space. A detailed analysis of the relationship between leakage power and temperature is presented in Chapter 2.

Current characteristics of devices significantly depend on changes in the supply voltage in all regions of operation. With a supply voltage droop of ΔV, both the gate and drain bias seen by the devices reduce, resulting in a reduction in the device current. This increases the path delays and causes non-critical paths to become critical. Further, it worsens signal slew, which can increase the short circuit power drawn and makes the circuits more susceptible to noise.

1.4 Temporal Variations

So far we have discussed how a FET's characteristics strongly depend on the fabrication processes and environmental factors. Once a chip is manufactured, packaged, tested (for correct functionality), and shipped to the customers, it is expected to function at the tested voltage, temperature, and frequency till the end of its usage life-time. Assumption is that if we do not have moving parts (as in mechanical machines) then machines do not age (analogous to human aging). However, physical changes can occur in a FET due to movement of charges – electrons and holes – and breaking of atomic bonds. Key mechanisms which affect a FET's behavior with time are:

(1) *Time-Dependent Dielectric Breakdown (TDDB)*: Creation and joining of defects in gate-dielectric causing gate dielectric breakdown.
(2) *Hot Carrier Injection (HCI)*: Defects in gate stack by highly energized carriers under large lateral (drain-to-source) electric fields causing shift in threshold voltage.
(3) *Bias-Temperature Instability (BTI)*: Capturing of holes (electrons) from the inverted channel in PFETs (NFETs) by the broken Si–H bonds (charge-trapping sites in high-k gate dielectrics such as HfO_2).

All the above mechanisms accelerate under large electric fields and high temperatures. In this section we discuss the mechanisms which *vary* the electrical characteristics of a circuit with usage. Many engineers term it as *degradation* of electrical characteristics; however, we will refrain from using the word degradation as in some aspects, electrical aging is not analogous to human or mechanical aging. There are several trade-offs at FET level which pose challenges in meeting desired circuit requirements. Say, if we want to increase performance, we will end up increasing power consumption as well. Therefore, a change in electrical characteristics which will degrade performance as well as power consumption will be a true degradation. We will learn that this is not always true when it comes to FET aging. In this section, we will learn different mechanisms which can affect circuit functionality with usage and also how we can estimate them. Before we proceed, we need to stress that "use" of a FET is a key factor in temporal variation of the characteristics of a FET. When a FET is subjected to voltage bias and operated under certain temperature, materials can degrade resulting in breaking of Si–H bonds causing interface traps, gradual breakdown of gate dielectric due to vertical electrical field, electrons/holes moving in unwanted directions under large lateral, and transversal fields.

1.4.1 Bias Temperature Instability (BTI)

Since the invention of semiconductor transistors, it has been widely known that high voltage and/or temperature stress on the gate electrode with SiO_2 as gate dielectric can change the flatband-voltage V_{fb} or the threshold-voltage V_{T} of MOS transistors [13]. This phenomenon is called Bias-Temperature-Instability (BTI). Several researchers have extensively studied the impact of positive as well as negative high voltage stresses on MOS FETs. With SiO_2 as the dielectric, the main focus was only on negative BTI (NBTI) which impacts PFETs. However, with the usage of HfO_2 as part of the gate dielectric to facilitate scaling, positive BTI (PBTI) which impacts NFETs has also become significant.

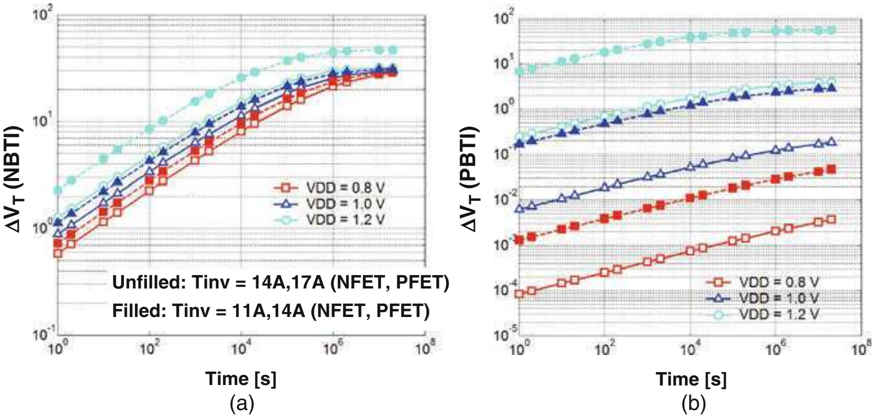


Fig. 1.12 Time-dependence of threshold voltage in TiN- and Re-gated devices with SiO2/HfO2 as dielectric stack due to (**a**) NBTI (on PFET), and (**b**) PBTI (on NFET). Two cases of inversion thicknesses (Tinv) are shown to analyze the impact of scaling *Source*: [15]

We saw that high temperatures cause $|V_T|$ to reduce, thereby giving a chance to have higher performance; however, mobility also reduces resulting in a conflict and eventually causing higher leakage power and lower performance. Interestingly (and conveniently too), most of the effects of device aging mechanisms can be understood by the change in V_T only. Figure 1.12 shows the time-dependence of threshold voltage shift due to NBTI and PBTI. Scaling the inversion layer thickness (T_{inv}) increases the vertical electric field at the given V_{DD} causing increase in V_T shifts due to BTI.

1.4.1.1 Negative-Bias-Temperature-Instability (NBTI)

NBTI relates to the instability induced in a FET due to generation of traps at FET channel and gate dielectric interface. A FET's channel is made of highly ordered crystalline silicon whereas gate dielectric is traditionally made of amorphous SiO_2. The interface surface of these two dissimilar materials is rough, resulting in dangling Si atoms from the channel due to unsatisfied chemical bonds. These dangling atoms are the interface traps which can lead to poor performance (lower on-state current) due to charge trapping and scattering. Hence, traditionally, the FETs are hydrogen annealed during fabrication after the formation of Si–SiO_2 interface. Hydrogen gas diffuses to the interface resulting in binding of hydrogen atoms (H) to the dangling Si atoms. Figure 1.13 shows the resulting Si–SiO_2 interface with hydrogen-passivated Si atoms. These Si–H bonds can break under large vertical negative electrical field ($V_{\text{GS}},\ V_{\text{GD}} < 0$) and high temperature. Dissociation of Si–H atoms will result in generation of interface traps. Note that when gate–drain and gate–source voltages are negative in a PFET, channel is inverted and conducting (whereas NFETs are off or under accumulation). Holes in PFET's inversion layer tunnel to SiO_2, and are captured by the Si–H bonds thereby weakening the bond.

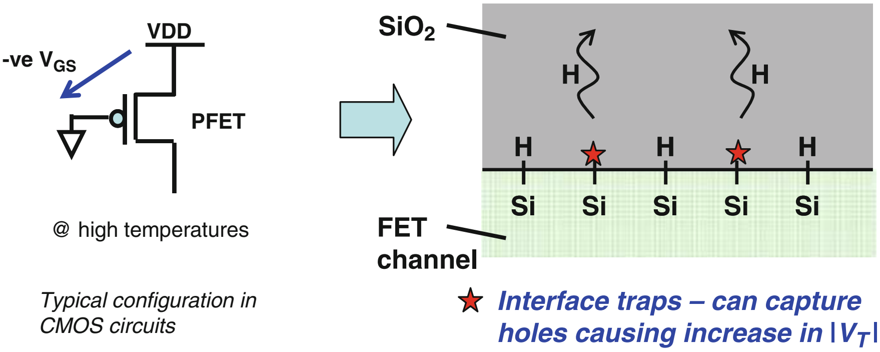

VDD
-ve V_{GS}
PFET
@ high temperatures
Typical configuration in CMOS circuits
SiO2
H
H
H
H
H
Si
Si
Si
Si
Si
FET channel
Interface traps – can capture holes causing increase in $|V_T|$

Fig. 1.13 Schematic of hydrogen dynamics after it is released from a Si–H bond. Interface traps are denoted by * [16]

Subsequently, high temperatures result in dissociation of Si–H pair resulting in one donor-like interface trap or Si dangling bond, and one H atom which diffuses in the oxide. These interface traps can capture the holes in an inverted PFET channel resulting in V_{GS} to become *extra* negative to reach similar strong inversion which existed without interface traps. Hence, V_T of a PFET will become more negative ($|V_T|$ will increase) resulting in reduced on-state current (keeping same V_{DD}). Note that such increase of $|V_T|$ in a PFET also reduces off-state leakage power, and effective capacitance thereby reducing active power. This change in V_T of a PFET is called NBTI-induced change in V_T.

In early (before the year 2000) technologies, power supply was reduced with gate dielectric scaling to maintain the near constant vertical electrical field. However, in recent technologies, power supply is scaled less aggressively compared to gate–dielectric to improve performance. This has resulted in increased vertical electric field. Further, power density is increasing due to packing of more functionality in same area with every technology resulting in increased local temperatures. Also, oxides are nitrided to prevent Boron penetration from polysilicon gate to the Si channel. Nitrogen is also added in gate dielectric to increase gate dielectric constant thereby increasing gate capacitance and performance. Unfavorably, it increases the NBTI-induced generation of interface traps [17].

Present day VLSI circuits operate at high temperatures (∼60–100°C) and also have large vertical electric field resulting in NBTI induced generation of interface traps in PFETs. The interface traps make the V_T of a PFET more negative and reduce carrier mobility. There are several physics models proposed by researchers; the most popular among them is reaction-diffusion model [18–24]. According to R-D model, inversion holes can break Si–H bonds freeing up H atoms (neutral) and creating donor like interface traps. H atoms can either diffuse away from the interface into the oxide or can anneal an existing trap. Interface traps result in increased V_T which can be given as

$$\Delta V_T(t) \approx (m_\mu + 1)\frac{qN_{IT}(t)}{C_{ox}} t^n \tag{1.3}$$

where m_μ is the mobility-degradation factor, N_{IT} is the number of interface traps at time t, and C_{ox} is oxide capacitance. The time-exponent n is $\sim$ 1/6 [22, 23] for on-the-fly measurements whereas it is >1/6 for delay-based measurements. Here, we will take $n = 1/6$ and include the recovery separately. Along with interface trap generation, other processes such as hole-trapping can also occur [25, 26] changing the form of above equation. Hence, in current technologies, change in V_{T} is empirically estimated by fitting the equation below

$$\Delta V_{\mathrm{T}}(t) = A.V_{\mathrm{stress}}^{k} \mathrm{e}^{-Ea/kT} t^{n} \tag{1.4}$$

where parameters A, k, n, and Ea (activation energy) are fitted to match measured data. V_{stress} is the stress voltage. Typically the chip testing time (approximately days–weeks) is significantly shorter than the desired lifetime ($\sim$ 10 years). Hence, accelerated characterization is performed at elevated stress voltages and temperatures, and the models are generated to predict the lifetime of a FET under operating voltage and temperatures.

Spatial Variation in NBTI

NBTI primarily depends on the number of interface traps. If a FET size ($L \times W$) is small, the number of traps can differ between FETs resulting in different NBTI degradation even at identical stress conditions. This difference in number of traps between FETs is observed to be random in nature and follow similar scaling law as RDF. Stewart Rausch [27] has given a semi-analytical model to quantify the statistics of V_{T} mismatch between the neighboring FETs in sub-100 nm technologies. The standard deviation of distribution of ΔV_{T} follows Poisson model (dependence on mean ΔV_{T} shift) and can be written as

$$\sigma_{\Delta} V_{\mathrm{T}} = \sqrt{\frac{2K_1 q T_{\mathrm{ox,eff}} \mathrm{Mean}(\Delta V_{\mathrm{T}})}{\varepsilon_{\mathrm{ox}} A_{\mathrm{G}}}} \tag{1.5}$$

where $T_{\mathrm{ox,eff}}$ is the effective oxide thickness of gate dielectric, $\varepsilon_{\mathrm{ox}}$ is the dielectric constant of $\mathrm{SiO_2}$, A_{G} is the gate area ($L \times W$) and mean (ΔV_{T}) is the mean shift in V_{T} as computed by (Equation 1.4). K_1 is an empirical parameter and found to be between 2.5 and 3.0 in most of the measured cases. Inclusion of NBTI statistics for analysis becomes important in small area FETs such as SRAM arrays.

1.4.1.2 Positive-Bias-Temperature-Instability (PBTI)

To reap the benefits of technology scaling, gate dielectric (primarily $\mathrm{SiO_2}$) thickness is aggressively scaled to increase gate capacitance, thereby increasing gate control over the channel. In sub-50-nm MOSFETs, due to short-channel effects, $\mathrm{SiO_2}$ needs to be scaled to $\sim$ 1 nm. Such thin gate oxides result in high gate direct tunneling leakage currents. Hence, the introduction of high-k gate dielectric became inevitable in sub-50-nm technologies. HfO_2 is one of the most promising candidates as high-k dielectric due to large dielectric constant 20–25, large band gap (5.6 eV), and thermal stability in contact with silicon.

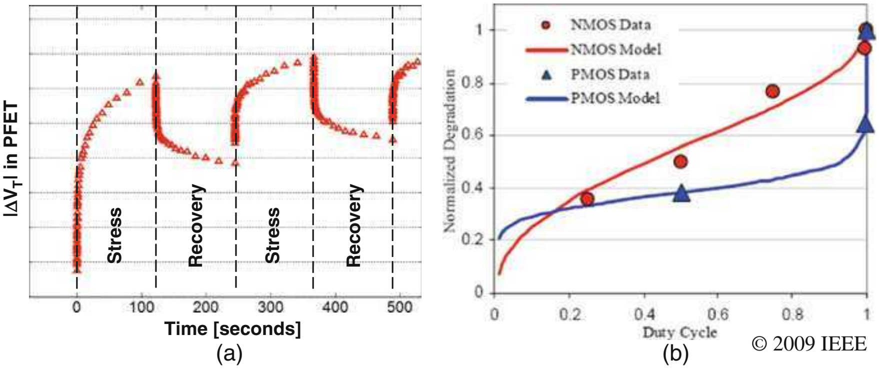
The V_T shift in MOSFETs with high-k dielectric is primarily attributed to charge trapping in high-k layer [28]. The trap density in high-k layer is much higher than SiO_2. Hence, at low stress voltages, V_T shifts due to filling of existing traps. Creation of traps is observed at high stress voltages. The reasons for significant charge trapping in high-k gate stacks can be attributed to process-induced defects, impurities, and material-related traps. There have been significant researches in last few years to understand the charge trapping phenomenon [29–32], deemed necessary for optimizing high-k dielectric for VLSI circuits. The ΔV_T due to PBTI is computed in similar fashion as for NBTI by empirically fitting the Equation (1.4).

1.4.1.3 BTI Recovery

As a FET is turned OFF, it relaxes thereby releasing the trapped charges. Traps can be broadly classified as slow and fast traps. Fast traps get relaxed quickly but slow traps release charges relatively slowly. They can also be classified as recoverable and permanent components of BTI-induced V_T shift [33]. Fraction Remaining (FR), after the stress has been removed, follows the universal recovery curve [34, 35].

$$\mathrm{FR} = \frac{1}{1+\alpha\left(\frac{T_{\mathrm{relax}}}{T_{\mathrm{stress}}}\right)^{n}} \tag{1.6}$$

where T_{relax} is the duration of relaxation and T_{stress} is the stress time. α is an empirical parameter (fitted separately for NBTI and PBTI) and n is the time exponent used in Equation (1.4). Figure 1.14a illustrates the ΔV_T with multiple stress-relaxation-stress cycles. Note that as soon as the stress is removed, recovery happens. Figure 1.14b shows the impact of duty cycle on the V_T degradation. Duty



© 2009 IEEE

Fig. 1.14 (**a**) PMOS Stress-Relax-Stress behavior. (**b**) Alternating OTF measurements for varying duty cycle. Duty cycle of 1 represents the static stress *Source*: [36]

cycle of 1 represents the static stress when the FETs are continuously ON. Duty cycle of 0.5 represents alternating stress causing FETs to be ON 50% of the time. It shows that in a circuit, if we can somehow make sure that a FET does not see static stress, we can increase the life of IC.

Typically we do accelerated stress at very high voltage (sometimes just below the breakdown limit) and then lower the voltage to measure the change in V_T. V_T is typically measured using constant drain current method (V_{GS} when I_{ds} is some pre-determined value). FET instantaneously recovers as soon as the gate voltage is lowered to measure V_T, making it difficult to measure the actual change in V_T. Several measurement techniques have been proposed to circumvent this. One popular method is on-the-fly (OTF) measurement. In OTF, we measure the drain current (let us say with $|V_{GS}| = V_{DD}$ and $|V_{DS}| = 50\,\text{mV}$ at time $t = 0$ and build a relationship $I_{dlin} = k(V_{DD}\text{-}|V_T|)^{alpha}$. Now, during measurement, instead of measuring V_T, we measure ΔI_{dlin} and back compute the ΔV_T. Further, recovery varies with the mechanism – NBTI or PBTI – as well as with varying gate stack [37]. Hence, as technology is changed, empirical fitting parameters in BTI-induced V_T shift equations may have to be retuned.

1.4.1.4 Estimation and Impact of BTI in Circuits

In the experiments, FETs are typically stressed using static stress (gate is always ON during stress) and alternating stress (gate is switching at different frequencies with varying duty cycle). These stresses can be conveniently used to generate a threshold voltage model dependent on stress time, frequency and duty cycle. However, in a circuit, more often each FET sees a unique gate signal during the life of usage. This signal depends on

- Applications Executed:

  For example, an application involving multiplication of 40-bit data on a 64-bit multiplier will result in the 24 most significant bits never switching. Another example can be clock signals which always switch irrespective of computation being done in logic blocks.

  Also, in a cache (say SRAM), it is quite possible that some bits are regularly flipped whereas some bits are read often but not flipped resulting in varying impact of BTI between bits. We will study the impact of BTI in SRAMs in more detail in the next section.
- Power Reduction Techniques Used – Dynamic Voltage Scaling (DVS), power–gating, etc.

  In a multi-core chip, all the cores are rarely used at maximum throughput. A core not heavily needed can be run at a lower frequency (by lowering the power supply voltage) to save power. This will result in lower BTI impact in this core. Further, if a core is completely shut down (say using power-gating), it will help in recovery.

There are recent efforts to build commercial tools [38], which can monitor the gate signal of each FET and apply suitable threshold voltage shift using pre-characterized models. This enables a designer to estimate the EOL shift in a circuit's performance, power, or robustness under real usage conditions. Other efforts are focused to build a BTI aware standard cell library [39].

From circuit application point of view, the nature and analysis of BTI can be broadly classified into two categories – impact of BTI in logic circuits (performance and power are major concerns) and memory circuits (stability and performance are major concerns). In this section, we will focus on these two separately.

BTI Impact in Logic Circuits

Critical metrics for a logic circuit are performance and power. As we studied earlier, BTI results in increased threshold voltage of NFETs and PFETs which can result in lower currents thereby reducing performance and leakage power dissipation. First let us analyze the impact of BTI on a static CMOS inverter. Figure 1.15 shows the output response of an inverter to an input voltage pulse. The output rise time primarily depends on the strength of PFET to charge the output load to V_{DD}, whereas, output fall time depends on the strength of NFET to discharge the output load. NBTI and PBTI will reduce the driving strengths of PFET and NFET, respectively, resulting in increased rise and fall delays. The change in V_T will depend on the duration for which a FET was ON. For example, let us consider two input voltage pulses as shown in Fig. 1.16. In both the cases, the input is high for the duration t_1 and low for the duration t_2. Let us look at the net FET degradation in both the cases computed by multiplying $\triangle V_T$ due to stress (Equation (1.4)) and fraction remaining (Equation (1.6)):

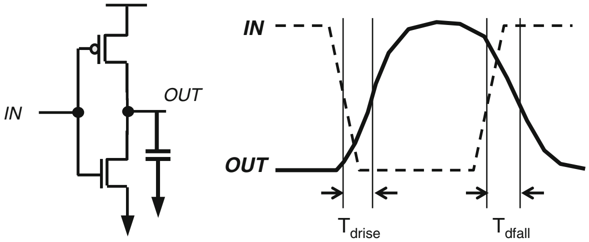


Fig. 1.15 Schematic and switching waveforms of an inverter

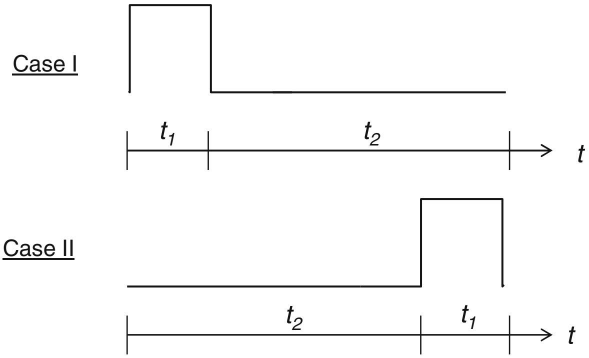


Fig. 1.16 Two different input signals to an inverter

Case – I:

$$\Delta V_{\mathrm{T}}(\mathrm{PFET}) = F_{\mathrm{NBTI}}(t_2)\ldots(\mathrm{a}) \quad \Delta V_{\mathrm{T}}(\mathrm{NFET}) = \frac{F_{\mathrm{PBTI}}(t_1)}{\left(1+\alpha_{\mathrm{PBTI}}\left(\frac{t_2}{t_1}\right)^{\beta_{\mathrm{PBTI}}}\right)}\ldots(\mathrm{b}) \tag{1.7}$$

Case – II:

$$\Delta V_{\mathrm{T}}(\mathrm{PFET}) = \frac{F_{\mathrm{NBTI}}(t_2)}{\left(1+\alpha_{\mathrm{NBTI}}\left(\frac{t_1}{t_2}\right)^{\beta_{\mathrm{NBTI}}}\right)}\ldots(\mathrm{a}) \quad \Delta V_{\mathrm{T}}(\mathrm{NFET}) = F_{\mathrm{PBTI}}(t_1)\ldots(\mathrm{b}) \tag{1.8}$$

Where $\mathrm{F_{NBTI}}$ and $\mathrm{F_{PBTI}}$ are the $\mathrm{V_T}$ shifts due to stress and computed using Equation (1.4). The FET degradation in the two cases is quite different. Hence, it is important to consider the actual signal at the gate instead of just looking in terms of duty cycle. If instead of looking at actual signal we had assumed that in the total time of t_1+t_2, NFET (PFET) is stressed for the duration of $t_1(t_2)$ and relaxed for the duration of $t_2(t_1)$, we would have wrongly estimated the same degradation in both the cases –(Equation (1.7b)) for NFET and (Equation (1.8a)) for PFET. That would have resulted in underestimation of PFET degradation in case I and underestimation of NFET degradation in case II. In practice, during EOL estimation via simulation, it can be quite difficult to monitor the input signal of each FET in a complex circuit. Computation complexity is traded with accuracy by averaging the signal over a long period of time. To date, no such efficient method exists which can accurately estimate the usage-based EOL degradation in each FET of a complex VLSI circuit.

The challenge in front of a circuit designer is to ensure reliable functionality of a circuit during the life of usage. If EOL shift can be accurately predicted during pre-Si circuit tuning, a sufficient guard-banding can be applied such as making extra slack available in critical paths. However, it can be quite difficult to test a circuit for all benchmark applications while monitoring the gate signal of each FET. Hence, on-chip BTI monitors are required to continuously monitor the frequency degradation in a chip and make self-repairs if necessary. The nature of self-repairs can be:

- In multi-core processors, shutting down a core for some time to do BTI recovery
- Power-gating the unused portions of a chip to do BTI recovery along with saving power
- Lowering voltage to reduce stress
- Using redundant copies of the degraded circuits, especially if the primary circuits become unusable due to HCI or TDDB (discussed in next section).

One other important aspect of analyzing BTI impact on a circuit is understanding the relative sensitivity of a circuit's performance to NBTI and PBTI. To a first order, the relative sensitivities of their drive current's strength to V_{T} can be simplified as

$$\frac{(\partial I_{\mathrm{DS}}/\partial V_{\mathrm{T}})\,|_{\text{pull-down}}}{(\partial I_{\mathrm{DS}}/\partial V_{\mathrm{T}})\,|_{\text{pull-up}}} = \left(\frac{\mu_{\mathrm{electron}}}{\mu_{\mathrm{hole}}}\right)\left(\frac{W_{\text{pull-down}}}{W_{\text{pull-up}}}\right) \tag{1.9}$$

Typically, electron mobility (μ_{electron}) in pull-down NFET is twice of hole mobility (μ_{hole}) in pull-up PFET and width of pull-up ($W_{\text{pull-up}}$) is designed to be higher than the width of pull-down ($W_{\text{pull-down}}$) to balance the rise and fall delays. For example, 50 mV of V_T shift in either PFET (due to NBTI) or NFET (due to PBTI) will result in approximately same amount of increase in RO frequency.

BTI Impact in Memory Circuits

BTI can be understood as increase in V_T with time. As we saw earlier that the impact of BTI can potentially be different for each FET in a circuit. Hence, the ratioed circuits which primarily depend on the relative driving strengths among different FETs can be severely affected. In this section, we focus on Static Random Access Memory (SRAM) to understand the impact of BTI [15, 40, 41].

Figure 1.17 shows the schematics of an SRAM cell under two conditions – static stress and alternating stress. Static stress happens when a cell is storing the same data for long period of time resulting in a cell becoming asymmetric, i.e., NL (due to PBTI) and PR (due to NBTI) become weak. Note that a cell will undergo static stress even if it is read several times or written with the same data several times. Due to static stress, gradually, the read operation will become unstable in this cell – reading "0" at left-node may cause failure as NL has become weak. We do not consider the impact of PBTI in access FETs (AXL and AXR) because they are turned on for short duration of time. However, in some cases when a cell is accessed very frequently, the access FETs may also become weak resulting in *reduced* impact of PBTI on read stability. Weaker pull-down NFET will also reduce the read current resulting in increased access time. Read current dictates how fast the BL discharges through the series combination of AXL and NL (for V_L= "0"). Since reading is getting unstable due to static stress, writing will become easier. For example, flipping the left node from "0" to "1" will become easier because NL has become weak whereas PL retains its original strength. Also flipping the right node from

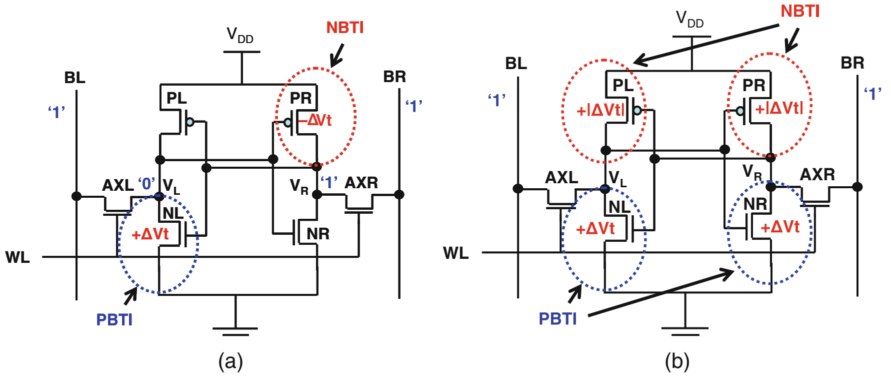


Fig. 1.17 Schematic of SRAM cell showing the FETs affected under (**a**) Static stress, and (**b**) Alternating stress

"1" to "0" has become easier because PR has become weak whereas NR retains its original strength. Hence, flipping data after long duration of static stress will become easier. On the other hand, a second write to *again flip the data* will become difficult [15].

Alternatively, a cell may be regularly flipped over the period of use resulting in the cell storing "0" and "1" for approximately same duration of time. This will keep the cell symmetric, i.e., same PBTI degradation in NL and NR, and same NBTI degradation in PL and PR (Fig. 1.17b). Considering negligible degradation in access FETs, the read will gradually become unstable and read access time will also increase (due to weak pull-down NFETs). Write operation will become easier, and unlike static stress, even second write operation will also be easier. Further, for a given duration of use, each FET will see the effective stress of half the use time and relaxation for the remaining half. On the other hand, static stress will cause the two FETs (NL and PR) to see the stress during whole use time without any recovery. Note that the static and alternating stress cases are the two extremes. Majority of cells in an SRAM array will see the stress somewhere in between.

In SRAM cells, width of pull-down is typically made higher than the width of pull-up to maintain read stability. Looking at (Equation (1.9)), let us assume that $\mu_{\text{electron}} \sim 2\mu_{\text{hole}}$ and $W_{\text{pull-down}} \sim 2.5W_{\text{pull-up}}$. Now, 50 mV of V_{T} shift due to PBTI will reduce the drive strength of pull-down NFET roughly 5*X* more than the similar V_{T} shift due to NBTI will do in PFET. Hence, the stability (read and write) of an SRAM cell is more sensitive to PBTI than NBTI.

1.4.2 Hot-Carrier-Injection (HCI)

Let us consider an NFET (as part of static CMOS inverter) during switching (Fig. 1.18). Its gate voltage rises turning it ON resulting in discharge of output node or drain. As the gate-to-source voltage rises, channel gets inverted resulting in abundance of electrons in the channel. In the initial part of switching, when drain-to-source voltage is high resulting in large lateral electric field, electrons gain high kinetic energy (called hot electrons) and accelerate toward drain. Near drain junction, the hot electrons generate secondary carriers through impact ionization. These primary and secondary electrons can gain enough energy to be injected into the gate stack resulting in the creation of traps at silicon/gate–dielectric interface and also bulk traps in the dielectric. These traps are electrically active and capture carriers at energy levels within the bandgap resulting in increased threshold voltage. HCI occurs in both NFETs and PFETs, however, it is prominent in NFETs because electrons face a smaller potential barrier than holes at the silicon/gate–dielectric interface. This aging mechanism has become less prominent in sub-50-nm technologies due to reduced operating voltages; however, it remains a concern due to high local electric fields. In addition, the dependence of HCI on stress time is higher than BTI bringing it at the forefront of aging culprits after long periods of usage.

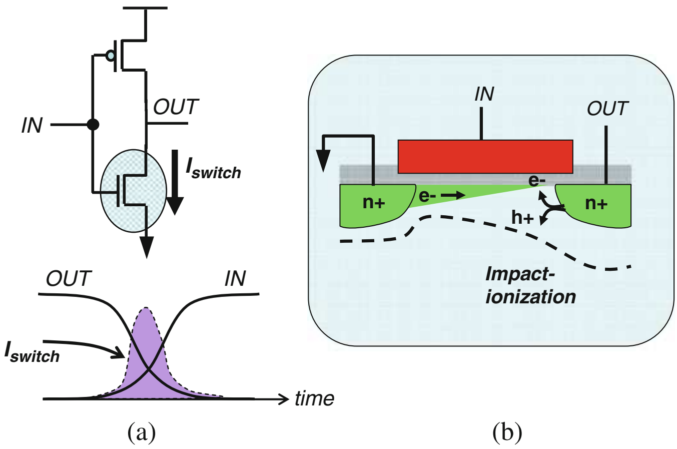


Fig. 1.18 (**a**) Schematic of an inverter showing switching current during input rising; (**b**) shows the mechanism of charged carriers causing impact ionization

The HCI occurs while a FET is switching contrary to BTI which is dependent on the gate voltage. HCI is empirically estimated in same way as BTI by empirically fitting (equation (1.4)). Authors in [42] measured HCI by separating it from BTI+HCI degradation.

1.4.3 Time-Dependent Dielectric Breakdown (TDDB)

As the name (TDDB) suggests, this phenomena pertains to gate dielectric breakdown under high vertical electric fields. Large voltage drop across the gate stack results in the creation of traps (or defects) within the gate dielectric. These defects may join together and form a conductive path through the gate stack causing dielectric breakdown. TDDB is an increasing concern as gate dielectric thicknesses are scaled down causing *faster* breakdown of gate–dielectric (less defects required to cause breakdown). The physical scaling of gate dielectric has been slowed down with the introduction of high-k dielectric materials; however, it is still a critical aging mechanism.

TDDB is measured by noticing the sudden jump in gate current. However, in today's sub-2-nm gate dielectrics, gate tunneling current is high enough to mask the jump in current. In TDDB analysis, mean-time-to-failure (MTTF) at a particular voltage bias is critical. In low-power circuits, the creation of first conductive or breakdown path through the gate dielectric seldom destroys the transistor. The lifetime of a FET after first breakdown typically depends on the spatial and temporal dependence of the subsequent breakdown [43, 44].

First order impact of TDDB in a circuit can be analyzed by connecting a resistor between gate–source and/or gate–drain in circuit simulations. Instead of a resistor, a constant current source can also be included to simulate the effect of gate current

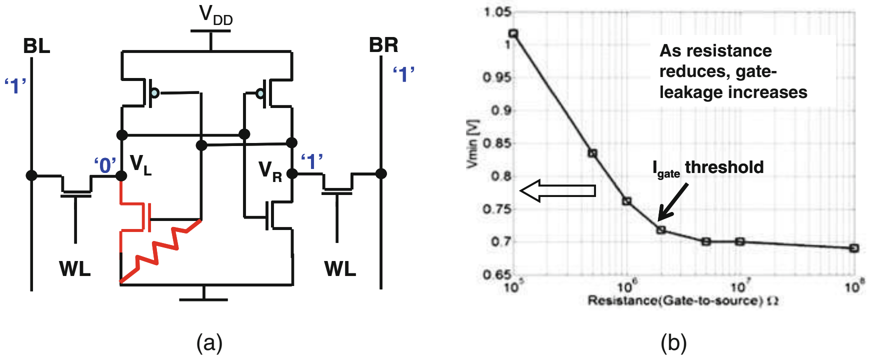


Fig. 1.19 (**a**) An SRAM cell with breakdown in left pull-down FET simulated by connecting a resistor. (**b**) Change in Vmin with resistance. Igate threshold represents the maximum gate leakage this cell can tolerate before suddent jump in Vmin

during breakdown. For example, Fig. 1.19 shows the simulation of gate–dielectric breakdown in the left pull-down FET of an SRAM cell by connecting a resistor between the gate and source. The value of resistance is varied to obtain the change in minimum operating voltage (V_{min}). This analysis can help a reliability engineer to obtain the gate current (I_{gate}) tolerance of a particular circuit.

1.5 Characterization Circuits

Characterization circuits play a vital role in understanding the sources of variation and their effects. The primary objective of such circuits (or sensors) is to evaluate the extent of variation between the characteristics of a particular device and its ideal expected behavior. In that sense, they play the role of on-chip monitors that are used to assess the health and malaise, if any, of the manufactured design. In earlier technologies, where the manufacturing process was well controlled, it would be sufficient to measure a handful of devices and determine its correctness with respect to the device models. But, with the increase in the extent of variation, the dynamic nature of environmental factors, and the time-dependent behavior of devices, it has become necessary to be able to periodically perform a statistically large number of measurements from monitoring circuits placed at different locations and environments across the die. These measurements help identify and isolate the various sources of variation and their impact on critical circuit parameters such as frequency of operation, noise immunity and power consumption. Further, characterization circuits can be used to determine the amount of guard-banding required during the design phase, and to drive adaptive compensation schemes post-fabrication in an attempt to improve the power and performance characteristics of the design. In this section, we look at several commonly used characterization circuits.

1.5.1 Accurate I–V Sensors

In its simplest form, a characterization circuit consists of a device having each of its terminals connected to probe pads such that its bias voltages can be controlled independently. The current–voltage characteristics of this device can then be measured under different voltage conditions and its electrical parameters such as threshold voltage and source–drain resistances can be estimated and compared with the model specifications. If several such devices can be accessed and measured, a statistical estimation of the extent of variation can be obtained. However, this requires each terminal of each device to be accessible through the use of many probe pads, which makes such an approach prohibitively expensive in terms of silicon area and infeasible.

A multiplexer-based approach can be used to overcome this problem, an example of which is shown in Fig. 1.20. [45]. Here, devices are placed in an $m \times n$ array. To measure the current characteristics of any device, a suitable bias is applied to

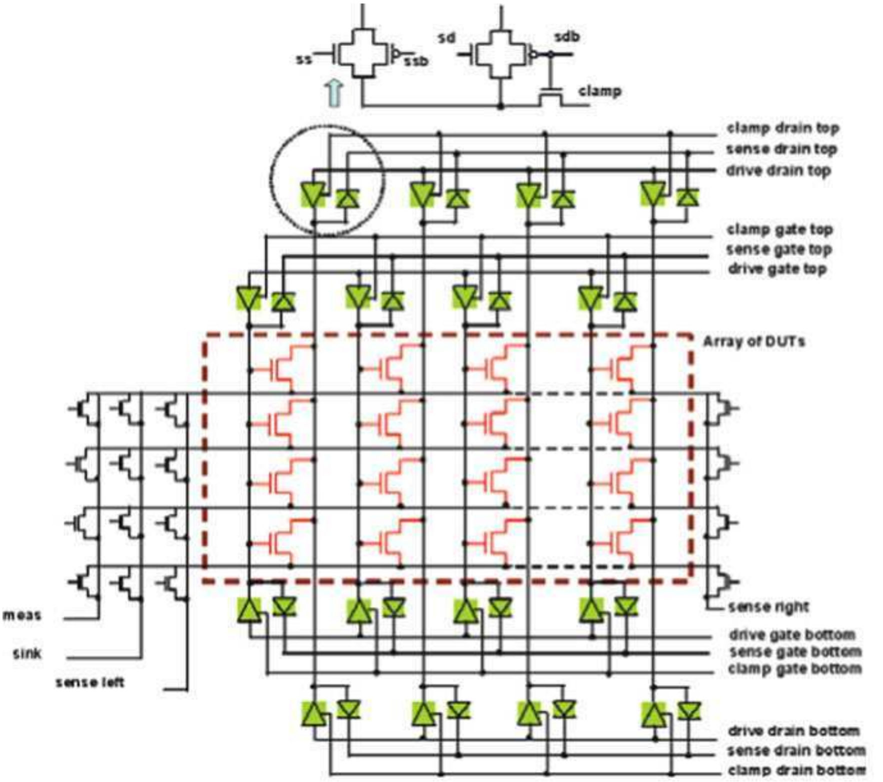


Fig. 1.20 Accurate I–V measurement sensor [45]. Each of the devices can be individually selected and its current–voltage characteristics measured

that particular row and column, while the rest of the rows and columns are clamped to an unselected state. The current of the selected device under test (DUT) is then measured through the *meas* terminal. To minimize noise effects, the leakage currents of all unselected devices are steered away through a secondary terminal (labeled *sink* in the figure). The impact of undesired voltage drops along the wires and through can be minimized by forcing the gate and drain terminals from both ends of the array. This reduces the IR drop by half and improves supply voltage integrity. In addition, sense nodes are introduced (sense left and sense right) that detect the actual voltage at these sense points, and the applied bias can be adjusted to cancel out any undesired voltage droops. The control of the select signals for the clamp circuits for each column (circled in Fig. 1.20) and the switches for enabling the *meas*, *sink,* and *sense* paths is accomplished through scan chains placed around the array of DUTs.

Devices of various physical dimensions (*L*, *W*), with different number and placement of contacts, physical design arrangements (isolated vs. densely packed), etc., can be seamlessly integrated into the array. This allows for a comprehensive comparison and evaluation of the different variations sources.

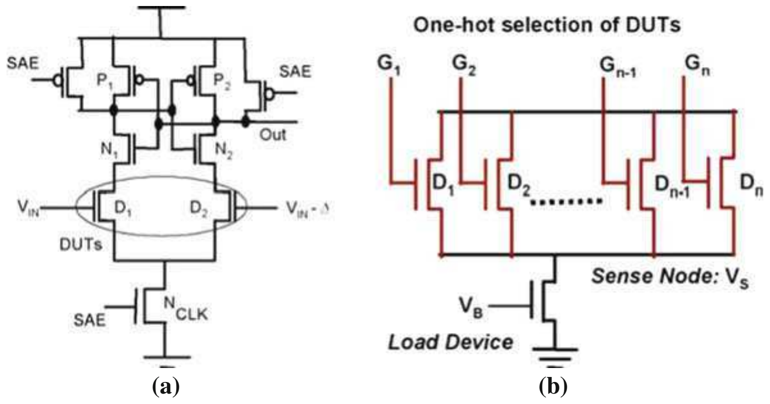
These I–V kind of sensors are very attractive due to their simplicity. The independent control of the terminal voltages allows great flexibility and complete characterization of the devices across the entire range of operation. The complete I–V measurement also gives a high level of confidence in the statistical measurements. However, they rely on accurate current measurements which require precision external current meter terminals (though on-chip current to voltage conversion circuits can be used) and the application of analog voltage at the device (these can also be generated on-chip but add complexity to the circuit). The characterization time is considerably large, since multiple measurements need to be made for each device.

1.5.2 Local Variation Sensors

The objective of local variation sensors is to estimate the mismatch between identical nearby devices without the need for complete I–V measurements. These sensors need to ensure that the effect of any global process or environmental variation is cancelled out. This is usually accomplished by using a differential measurement approach. Local variation sensors can be broadly classified into two sub-categories based on the number of devices that are compared against each other.

1.5.2.1 Mismatch Sensors

This type of process sensors detects the mismatch between two identical biased devices using a differential circuit configuration. The sensor is designed such that the output is strongly dependent on the mismatch between the DUTs. During the design of these sensors, it is important to ensure that any mismatch from the non-DUT devices in the differential configuration does not impact the overall measurement accuracy.



© 2008 IEEE © 2009 IEEE

Fig. 1.21 Sensors for local variation detection (**a**) Mismatch based sensor [46] (**b**) Array based sensors [47]

One example of such a circuit is shown in Fig. 1.21a [46]. A current latch sense amplifier (CLSA) circuit is used as the differential configuration, where D_1 and D_2 are the DUTs whose mismatch is estimated. The offset voltage for correct sensing of the CLSA depends on the mismatch between the DUTs. This offset voltage is measured by varying the gate bias of one of the two devices (D_1 is biased at V_{in} while D_2 is biased at $V_{\text{in}} - \Delta$) and determining the minimum Δ required for correct transition at the output. The measurement resolution is determined by the granularity of Δ which can either be applied externally or generated on-chip. A mismatch in the output driver portion of the CLSA can introduce some error into the measurement.

1.5.2.2 Array Sensors

Array sensors aim to generate a voltage that is a direct representation of the current (and hence device threshold voltage) in a device selected from an array of devices [48–51]. This sense voltage is then polled for different selections from the array of DUTs, and the distribution of the sense voltage indicate the extent of local random variations.

Figure 1.21b illustrates an example of such a characterization circuit [47]. An array of individually selectable DUTs is arranged in a stacked configuration with a common load device (LD). A first DUT, $D_{1,}$ is selected. The voltage at the sense node V_{s} depends on the relative strength of D_1 and LD. The sense node voltage changes when this D_1 is deselected and a second DUT D_2 is selected, since it now depends on the strength of D_2 and LD. This change in sense node voltage is a representation of the difference between the current characteristics of the two DUTs. All the DUTs can be selected in a cyclical order, and the statistical characteristics of the sense node voltage is computed to determine the extent of local variation.

1.5.3 Weak Inversion Sensors

As the name suggests, these sensors consists of DUTs being biased in the sub-threshold region [52, 53]. In the sub-threshold region, the current through the device is an exponential function of threshold voltage, temperature, and supply voltage (due to drain-induced barrier lowering), and hence is extremely sensitive to variation. The principle of operation of these sensors is shown in Fig. 1.22a. The current from a constant current source is fed through a DUT biased in weak inversion, and the voltage at the intermediate node (V_{out}) is measured. If the device has a higher threshold voltage than expected, the sub-threshold leakage for a given drain voltage is low. Hence, the drain bias voltage has to increase to cause the leakage current to increase (due to the DIBL effect) and match the reference current. Thus, V_{out} is higher than nominal. Similarly, V_{out} is lower if the DUT has a lower than expected threshold voltage.

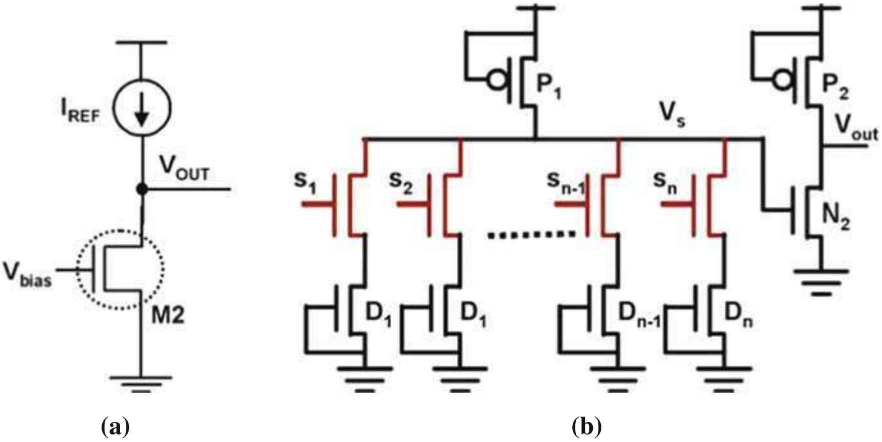


Fig. 1.22 Weak inversion based sensors (**a**) Principle of operation (**b**) An implementation [53]

An example of a weak-inversion sensor is shown in Fig. 1.22b [53]. Here, an array of DUTs (D_1 to D_n) is biased in sub-threshold region by grounding their gate terminals. High threshold voltage thick oxide device switches (S_1 to S_n) are used to select each DUT individually. Let us consider the case where $D1$ is selected, by setting $S_1 =$ “1” and S_2 to $S_n = 0$. The voltage at the intermediate node (V_{s}) is then a function of the sub-threshold leakage current of D_1 and the load device P_1. With the selection of a D_2, the sense node voltage changes depending on the mismatch between D_1 and D_2.

The second stage consisting of P_2 and N_2 acts as an amplifier and improves the sensitivity of the sensor. This is achieved by sizing P_1 such that the sense node voltage V_{s} biases N_2 in sub-threshold region in the absence of variation. A small change in the sense node voltage will result in an exponential change in the current through the second stage, thereby causing a larger change in the output voltage

(V_{out}). Weak inversion-based sensors are often used as temperature-monitoring circuits, due to the high sensitivity of current to temperature in the sub-threshold or diode-connected configurations.

Another class of sensors are diode sensors, in which the device under test is configured in a diode-configuration by tying the gate and drain of the device (instead of being biased in cut-off region). The readers are encouraged to refer to [54–57] for detailed illustrations.

1.5.4 Ring Oscillator Sensors

Ring oscillator-based sensors are the most popular category of sensors. They characterize variation based on delay through a chain of N gates with the output of the last gate tied back to the input to form a ring. The delay of each gate is a function of the switching characteristics of the devices which varies with the global process parameters and environmental conditions. Since the measured frequency is determined by the sum of *N* gate delays, these sensors capture global effects and the impact of any local variations is typically averaged out (for sufficiently large value of *N*).

In its simplest form, a ring oscillator-based sensor consists of an odd number of inverter stages (with the theoretical minimum for the number of stages in the ring oscillator being 3) [58, 59], as shown in Fig. 1.23. The frequency of oscillation of the ring is then measured to determine the extent of variation. A higher frequency indicates a fast – leaky design, while a lower frequency of implies a slow, timing critical design. This frequency can be measured externally using a high-precision oscilloscope. Typically, one of the stages of the oscillator is modified to introduce a clamp signal. When the clamp signal is asserted, oscillations are quenched and ring is clamped to a particular state at each of its nodes. This can be implemented by replacing one of the inverter stages in the ring with a Nand2 or Nor2 configuration.

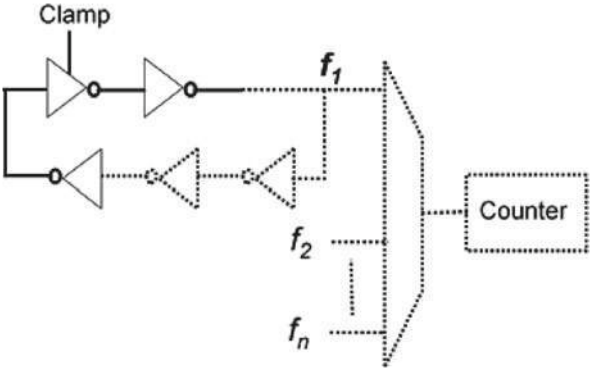
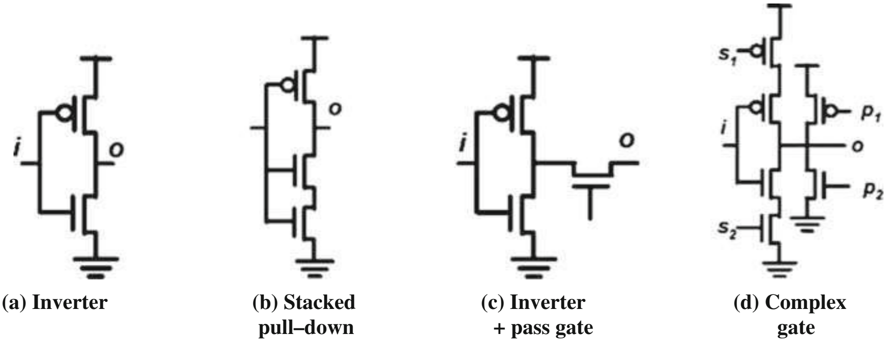


Fig. 1.23 Ring Oscillator Sensor consists of a chain of gates arranged in a ring. The output of several ring oscillators can be multiplexed and the frequency observed externally or detected on-chip using a counter

Frequency measurement is very attractive since it is independent of any additional delay stages that may be introduced between the output of the ring and the measurement point. This allows for multiplexers to be easily introduced at the output to allow for selection for one of many ring oscillators. Optionally, a counter with a fixed time base can be used to count the number of oscillations of the ring oscillator to get a digital representation of the shift in frequency.

While the Fig. 1.24a shows an inverter as the basic gate, they can be replaced by complex gates. As an example, many ring oscillators can be designed, with each having one of the base cells shown in Fig. 1.24. During characterization, each of the ring oscillators can be selected and its frequency is measured. By studying the spread in frequency of these oscillators, the impact of variations on different classes of circuits can be evaluated. For instance, base cell (b) is made of stacked pull-down trees, and hence is more sensitive to any global shift in NMOS threshold voltages. Similarly, base cell (d) consists of a pass gate in series with an inverter as the base cell and hence can be used to characterize variations in linear mode by suitably biasing the pass gate. In a more generic implementation, each of the stages of the ring oscillator can be a different gate. The only requirement in a ring oscillator-based sensor is that the total number of inversions around the loop be an odd number.



(a) Inverter **(b) Stacked pull–down** **(c) Inverter + pass gate** **(d) Complex gate**

Fig. 1.24 Different base cells can be used to implement Ring Oscillator Sensors to help isolate the impact of different variation sources (**a**) Inverter (**b**) Stacked pull-down (**c**) Inverter + pass gate (**d**) Complex gate

1.5.4.1 Characterization of Temporal Degradation

Ring oscillator-based sensors can also be used to characterize temporal variations effects [60–62]. By design, the various nodes in the ring oscillator are at logic low and high states for equal duration of time, thereby degrading both NFET and PFET devices. The ring oscillator frequency can be monitored continuously and any degradation in frequency with time can be attributed to device aging affects such as BTI and HCI (described in section 1.4). If device degradation under DC stress condition needs to be characterized, the clamp signal can be asserted to force the ring oscillator nodes into a particular *stress* state. The ring oscillator then sits in the non-switching state and degrades primarily due to BTI. After any duration of time t, oscillations can be resumed and the frequency is measured. This sequence of *stress* and *measurement* can be repeated at any desired periodicity. The degradation thus measured is a combination of both NBTI and PBTI, and some HCI effects. These effects can be isolated by an intelligent design of the base cell and control circuitry. For instance, the ring oscillator configuration (d) in Fig. 1.24 can be used to isolate NBTI and PBTI. The readers are encouraged to refer to [61] and [62] for a detailed analysis of such configurations.

1.5.4.2 Improving Ring Oscillator Sensitivity by Using On-Chip Reference

While characterizing the impact of temporal variations, it is important to cancel out the effect of any global variations. This can be achieved by using an on-chip reference ring oscillator which is placed in close proximity to the test ring oscillator. The physical proximity of these oscillators ensures that they experience similar global and environmental variations. The reference ring oscillator is not stressed by placing it on a separate supply voltage which is reduced during the stress phase, and hence does not undergo any degradation. In other words, the reference ring oscillator always exhibits its inherent post-fabrication behavior. During measurement, the degraded frequency of the test ring oscillator is compared against the reference ring oscillator. Figure 1.25 illustrates two ways of performing this comparison.

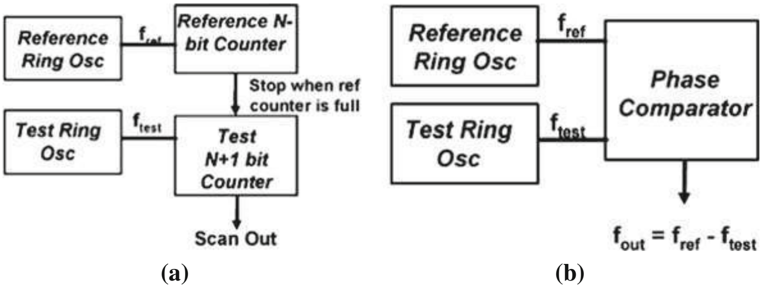


Fig. 1.25 The sensitivity of ring oscillator based sensors can be improved by using on-chip reference oscillator along with phase comparators on counter circuits

Counter-Based Comparator: In a counter-based comparison scheme, the reference ring oscillator sets up a time base that is equivalent to N times its period of oscillation, i.e., $T = 2^N/f_{\text{ref}}$. A second counter, counts the number of oscillations of the test counter during this fixed time duration. When the reference counter hits its maximum value, its stops the count in the test counters. The output of the test counter is now a digital representation of the frequency of the test ring oscillator relative to the reference ring oscillator.

Phase Comparator: A phase comparator generates an output frequency which is equal to the difference in the frequencies of the reference and the test counter. This output signal beats based on the time required for the two oscillators, which are both free running, to align. The beat frequency can then be fed to a counter or measured externally. This differential measurement results in a very high resolution and hence can be used to detect very small amount of degradation in device characteristics.

1.5.5 Path Delay Sensors

Path-delay sensors consist of a chain of gates, with a single transition being launched at the start of this chain. The number of gates traversed by this transition in a given clock period is determined. Path based sensors typically consist of latches and exclusive OR (XOR) gates that act as edge detection circuitry. Fig. 1.26 shows one such implementation. A series of latches can be added to the output of the XOR gates. The sensor is designed such that the transition travels through half the detection chain under nominal conditions in a given clock (ϕ) period. The location of the edge is detected by comparing the latch output with the expected logic value at that stage of the chain using exclusive OR gates and locating a "1– 0" change in the output. If the global process has shifted toward the fast corner, the delay of each of the gates will be small and the transition will traverse through a larger than nominal number of gates in the given clock period. On the other hand, in the presence of a global shift toward a slow corner, the number of gates traversed will be lower than nominal. Since the launched edge is racing against the clock, it is likely that some of the latches fall into the meta-stability problem resulting in incorrect sampling. A bubble rejection circuitry that detects an isolated "1" or "0" can be used to eliminate that problem.

The resolution of measurement is determined by the gate delay of the elements in the edge-detection circuitry (a fan-out of one inverter delay in this example) and hence is better than ring oscillator-based circuits. The sensing speed is also superior,

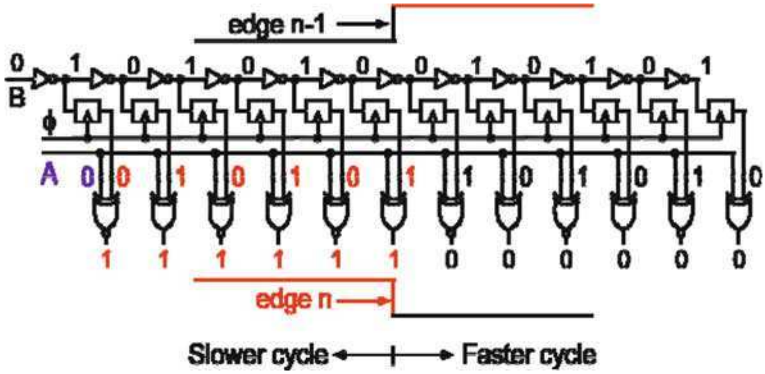


Fig. 1.26 Path-based sensor consisting of latches and exclusive ORs acting as edge detection circuit [63]

since an input transition can be launched in every cycle. This enables cycle-to-cycle variation to be detected using this type of sensor. For instance, if the activity of a neighboring circuit results in a current surge, and a resultant supply voltage droop, the delay of all the gates in the path will increase, and hence the edge will traverse through a fewer number of gates. By monitoring the output of the edge detector every cycle, the amplitude and duration of the supply voltage droop can be characterized. Path delay-based sensors can also be used to characterize temporal variation effects such as bias temperature instability and hot carrier effects [64]. The type of stress (static vs. alternating) and the duration can be controlled by the signal applied at the input of the chain.

While we have described some commonly used sensors, this is by no means an exhaustive categorization of characterization circuits. In certain scenarios, it becomes necessary to design on-chip sensors that detect more complex circuit parameters. Examples of these are SRAM variation monitors [65], noise margin detectors [66], N-P mismatch detectors [67] etc.

1.6 Concluding Remarks

Variations in sub-nanometer technologies have emerged as a critical obstacle to the ability to ship products that meet the system performance, power, and reliability requirements. This trend is expected to worsen in future technologies with newer sources of variation, increased power densities on-chip and bigger chips with larger thermal and supply gradients. Increased thermal profiles and the use of high-k metal gate technologies are also expected to intensify device degradation mechanisms resulting in an increased failure rate in the field. On-chip, characterization circuits that help in understanding these numerous sources of variations, their dependencies, and impact on different circuit styles and parameters are becoming necessary and prevalent in current designs. In addition to proper guard-banding, designs are likely to require intelligent autonomic systems with self-calibration and compensation schemes that help counter the ill-effects of these variation sources and enhance parametric yield.

References

1. Yu P, Shi SX, Pan DZ (2006) Process variation aware OPC with variational lithography modeling. DAC 785–790
2. Heng FL, Lee J-f, Gupta P (2005) Towards through-process layout quality metrics. Proc SPIE 5756:161–167
3. Bansal et al (2009) Yield estimation of SRAM circuits using virtual SRAM fab. ICCAD 631–636
4. Ahsan I et al (2006) RTA-driven intra-die variations in stage delay, and parametric sensitivities for 65 nm technology. VLSI Tech Sym 170–171
5. Wang C-C et al (2009) Modeling of layout-dependent stress effect in CMOS design. ICCAD 513–520

6. Sleight J.W et al (2006) Challenges and opportunities for high performance 32 nm CMOS technology. IEDM 1–4
7. Hane M et al (2003) Coupled atomistic 3D process/device simulation considering both line-edge roughness and random-discrete-dopant effects. SISPAD 99–102
8. Zhao W et al (2009) Rigorous extraction of process variations for 65-nm CMOS design. TED 196–203
9. Frank DJ et al (1999) Monte carlo modeling of threshold variation due to dopant fluctuations. VLSI Tech Sym 169–170
10. Asenov A, Kaya S, Brown AR (May 2003) Intrinsic parameter fluctuations in decananometer MOSFETs introduced by gate line edge roughness. IEEE TED 1254–1260
11. Asenov A, Jaraiz M, Roy S, Roy G, Adamu-Lema F (2002) Integrated atomistic process and device simulation of decananometre MOSFETs. International conference on simulation of semiconductor processes and devices (SISPAD 2002), Tokyo, pp 87–90
12. Franch R, et al (2008) On chip timing uncertainty on IBM microprocessors. In: Proceedings of the international test conference, Santa Clara, CA, pp 1–7, October 2008
13. Deal BE, Sklar M, Grove AS, Snow EH (1967) Characteristics of the surface-state charge (Q) of thermally oxidized silicon. J Electrochem Soc 114:266–274
14. Shiono N, Yashiro T (1979) Surface state formation during long-term bias-temperature stress aging of thin SiO2-Si interfaces. Jpn J Appl Phys 18:1087–1095
15. Bansal A et al (2009) Impacts of NBTI and PBTI on SRAM static/dynamic noise margins and cell failure probability. J Microelectron Reliab 642–649
16. Küflüoglu H, Alam MA (May 2007) A generalized reaction–diffusion model with explicit H–H2 dynamics for negative-bias temperature-instability (NBTI) degradation. IEEE Trans Electron Devices 1101–1107
17. Kimizuka N, Yamaguchi K, Iniai K, Iizuka T, Liu CT, Keller RC, Horiuchi T (2000) NBTI enhancement by nitrogen incorporation into ultrathin gate oxide for 0.10-pm gate CMOS generation. In: Symposium on VLSI Technology, pp 92–93
18. Jeppson KO, Svensson CM (1977) Negative bias of MOS devices at high electric fields and degradation of MNOS devices. J Appl Phys 48(5):2004–2014
19. Alam MA (2003) A critical examination of the mechanics of dynamic NBTI for PMOSFETs. In: IEDM technical digest, pp 346–349
20. Chen G, Chuah KY, Li MF, Chan DSH, Ang CH, Zheng JZ, Jin Y, Kwong DL (2003) Dynamic NBTI of PMOS transistors and its impact on device lifetime. In: Proceedings of the IEEE international reliability physics symposium, pp 196–202
21. Islam AE, Kufluoglu H, Varghese D, Mahapatra S, Alam MA (2007) Recent issues in negative-bias temperature instability: initial degradation, field dependence of interface trap generation, hole trapping effects, and relaxation. IEEE Tran Electron Devices 54(9)
22. Varghese D, Saha D, Mahapatra S, Ahmed K, Nouri F, Alam MA (2005) On the dispersive versus Arrhenius temperature activation of NBTI time evolution. In: IEDM Technical Digest, Washington, DC, pp 684–687
23. Krishnan A et al (2005) Material dependence of hydrogen diffusion: implications for NBTI degradation. In: IEDM technical digest, 688–691
24. Küflüo˜glu H, Alam MA (2004) A geometrical unification of the theories of NBTI and HCI time-exponents and its implications for ultrascaled planar and surround-gate MOSFETs. In: IEDM technical digest, San Francisco, CA, p 113
25. Mahapatra S, Ahmed K, Varghese D, Islam AE, Gupta G, Madhav L, Saha D, Alam MA (2007) On the physical mechanism of NBTI in silicon oxynitride p-MOSFETs: can difference in insulator processing conditions resolve the interface trap generation versus hole trapping controversy? In: Proceedings of the IEEE international reliability physics symposium
26. Yang T, Shen C, Li MF, Ang CH, Zhu CX, Yeo YC, Samudra G, Rustagi C, Yu MB, Kwong DL (November 2005) Fast DNBTI components in p-MOSFET with SiON dielectric. IEEE Electron Device Lett 26(11):826–828

27. Rauch SE III (December 2002) The statistics of NBTI-induced VT and beta mismatch shifts in pMOSFETs. Trans Dev Mat Rel 89–93
28. Gusev EP et al (2001) Ultrathin high-K gate stacks for advanced CMOS devices. In: IEDM technical digest, Washington, DC, 451–454
29. Zhang JF, Eccleston W (1998) Positive bias temperature instability in MOSFET's. Trans Electron Device 116–124
30. Zafar S, Callegari A, Gusev E, Fischetti MV (2003) Charge trapping related threshold voltage instabilities in high permittivity gate dielectric stacks. J Appl Phys 9298–9303
31. Zafar S (2006) A comparative study of NBTI and PBTI (charge trapping) in SiO2/HfO2 stacks with FUSI, TiN, Re gates. Symposium on VLSI Technology
32. Onishi K et al (June 2003) Bias-temperature instabilities of polysilicon gate HfO2 MOSFETs. Trans Electron Device 1517–1524
33. Grasser T et al (2007) Simultaneous extraction of recoverable and permanent components contributing to bias-temperature instability. IEDM Technical Digest 2007, pp 801–804
34. Rangan S, Mielke N, Yeh ECC (2003) Universal recovery behavior of negative bias temperature instability. In: Proceedings of the IEEE IEDM 2003, pp 341–344
35. Reisinger H, Blank O, Heinrigs W, Mühlhoff A, Gustin W, Schlünder C (2006) Analysis of NBTI degradation- and recovery-behavior based on ultra fast VT-measurements. In: Proceedings of the IEEE international reliability physics symposium (IRPS), Dallas, TX, pp 448–453
36. Ramey S, Prasad C, Agostinelli M, Pae S, Walstra S, Gupta S, Hicks J (2009) Frequency and recovery effects in high-κ BTI degradation. IRPS, pp 1023–1027
37. Denais M (2004) Interface trap generation and hole trapping under NBTI and PBTI in advanced CMOS technology with a 2-nm gate oxide. IEEE Trans Device Mater Reliability 715–722
38. Liu Z, McGaughy BW, Ma JZ (2006) Design tools for reliability analysis. In: Design automation conference, San Francisco, CA, 182–187, July 2006
39. Kumar SV, Kim CH, Sapatnekar SS (2007) NBTI-aware synthesis of digital circuits. In: Design Automation Conference (DAC), San Diego, CA, pp 370–375
40. Kang K et al (2007) Impact of negative-bias temperature instability in nanoscale SRAM array: modeling and analysis. TCAD, IEEE Transactions on Computer Aided Design, pp 1770–1781
41. Lin JC et al (2007) Time dependent Vccmin degradation of SRAM fabricated with high-k gate dielectrics. In: International Reliability Physics Symposium (IRPS), pp 439–444
42. Kim T-h, Wang X, Kim CH (2010) On-chip reliability monitors for measuring circuit degradation. J Microelectron Reliability, 1039–1053
43. Alam MA, Weir BE, Silverman PJ (2002) A study of examination of physical model for direct tunneling current in soft and hard breakdown – part II: principles of area, thickness, and voltage scaling. IEEE Trans Electron Devices 49:239–246
44. Alam MA, Weir BE, Silverman PJ (2002) A study of soft and hard breakdown – part I: analysis of statistical percolation conductance. IEEE Trans Electron Devices 49:232–238
45. Agarwal K, Liu F, McDowell C, Nassif S, Nowka K, Palmer M, Acharyya D, Plusquellic J (2006) A test structure for characterizing local device mismatches. In: Symposium on VLSI Circuits, pp 67–68
46. Mukhopadhyay S, Kim K, Jenkins K, Chuang C, Roy K (2008) An on-chip test structure and digital measurement method for statistical characterization of local random variability in a process. J Solid State Circuits 43(9):1951–1963
47. Rao R, Jenkins K, Kim J (2009) A local random variability detector with complete digital on-chip measurement circuitry. J Solid State Circuits 44(9) 2616–2623
48. Klimach H et al (2004) Characterization of MOS transistor current mismatch. In: Symposium on integrated circuits syst design, 33–38
49. Wang V, Shepard K (2007) On-chip transistor characterization arrays for variability analysis. Electron Lett 43(15):806–806

50. Drego N, Chandrakasan A, Boning D (2007) A test-structure to efficiently study threshold-voltage variation in large MOSFET arrays. In: International symposium on quality electronic design, San Jose, CA, 281–286
51. Terado K, Eimitsu M (2003) A test circuit for measuring MOSFET threshold and voltage mismatch. In: International conference on microelectronics test structures, pp 227–231
52. Kim C et al (2005) Self calibrating circuit design for variation tolerant VLSI systems. In: International online listing symposium, 100–105
53. Meterellyioz M, Song P, Stellari F, Kulkarni J, Roy K (2010) Characterization of random process variations using ultralow-power, high-sensitivity, bias-free sub-threshold process sensor. IEEE Tran Circuits Syst 99:1838–1847
54. Syal A. Lee V, Ivanov A, Altel H (2001) CMOS differential and absolute thermal sensors. In: International online listing symposium, 127–132
55. Chen P, Chen C, Tsai C, Ku W (2005) A time-to-digital-converter-based CMOS smart temperature sensor. J Solid State Circuits 40(8):1642–1648
56. Szekely V, Marta C, Kohari Z, Rencz M (2007) CMOS sensors for on-line thermal monitoring of VLSI circuits. IEEE Trans VLSI Syst 5(3)270–276
57. Chen Q, Meterelliyoz M, Roy K (2006) A CMOS thermal sensor and its application in temperature adaptive design. In: International symposium on quality electronic design, 248–253
58. Zhou B, Khouas A (2005) Measurement of delay mismatch due to process variations by means of modified ring oscillators. Int Sympo Circuits Syst 5:5246–5249
59. Bhushan M, Ketchen M, Polonsky S, Gattiker A, (2006) Ring oscillator based technique for measuring variability statistics. In: International conference on microelectronics text structure, 87–92
60. Karl E, Singh P, Blaauw D, Sylvester D (2008) Compact In-Situ sensors for monitoring negative bias temperature instability effect and oxide degradation. In: International conference on solid state circuits, 410–413
61. Keane J, Wang V, Persaud D, Kim C (2010) An all-In-One silicon odometer for separately monitoring HCI, BTI and TDDB. J Solid State Circuits 45:817–829
62. Kim J, Rao R, Mukhopadhyay S, Chuang C (2008) Ring oscillator circuit structures for measurement of isolated NBTI/PBTI effects. In: International conference on integrated circuit design and Technology, 163–166
63. Drake A. Senger R, Deogun H, Carpenter G, Ghiasi S, Nguyen T, James N, Floyd M, Pokala V (2007) A distributed critical-path timing monitor for a 65 nm high-performance microprocessor. In: International conference on solid state circuits, 398–399
64. Saneyoshi E, Nose K, Mizuno M (2010) A precise tracking NBTI-degradation monitor independent of NBTI recovery effect. In: International conference on solid state circuits, 192–193
65. Bhavnagarwala A, Kosonocky S, Radens C, Chan Y, Stawiasz K, Srinivasan U, Kowalczyk S, Ziegler M (2008) A sub-600-mV, fluctuation tolerant 65-nm CMOS SRAM array with dynamic cell biasing. J Solid State Circuits 43(4):946–955
66. Mojumder N, Mukhopadhyay S, Kim J, Chuang C, Roy K (2010) Self-repairing SRAM using on-chip detection and compensation. IEEE Trans VLSI Syst 75–84
67. Ghosh A, Kim J, Rao R, Chuang C (2008) On-chip process variation detection using slew-rate monitoring circuit. In: International conference on VLSI design, 143–149

Chapter 2
Power Dissipation

Wei Zhang, James Williamson, and Li Shang

Abstract This chapter discusses the power consumption issue of the mainstream CMOS technologies. During the past two decades, power dissipation has stood out as the foremost design challenge for general-purpose and application-specific integrated circuits (ICs). Considering and optimizing the circuit power efficiency has become essential. IC power modeling, analysis, design-time optimization, and run-time management techniques have been intensively studied. This chapter covers the basics of the IC power consumption issue. It first investigates the sources of IC power dissipation, and then discusses recent techniques for IC power analysis. Finally, it studies recently proposed power optimization techniques from circuit and physical design to system synthesis.

2.1 Introduction

Technology trends and design constraints drive the evolution of integrated circuit (IC) and system design. For decades, electronic systems have benefited from the relentless progress of semiconductor fabrication technology governed by Moore's Law. With each technology generation, ICs become denser, faster, and cheaper. However, with increasing technology scaling and system integration, IC power consumption has become a major design challenge, introducing a variety of power-related design issues.

Low-power IC design has been an active research field over the past 20 years. From modeling and analysis to design-time optimization and run-time management, IC power consumption issues have been thoroughly studied and carefully optimized. The existing large body of low-power IC research has effectively alleviated the IC power consumption challenges, and the semiconductor technology is thus allowed

L. Shang (✉)
Department of Electrical, Computer, and Energy Engineering, University of Colorado, Boulder, Co, USA
e-mail: li.shang@colorado.edu

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_2,

to continue to scale. For instance, from Intel Pentium 4 uni-core microprocessor to Intel Core i7 quad-core design, the microprocessor thermal design power remains approximately the same, but the overall system performance has been improved substantially. However, IC power consumption challenges will continue to grow. As projected by International Technology Roadmap for Semiconductors (ITRS) [1], power will remain to be a limiting factor in future technologies. The power optimization techniques, which have been widely deployed in the past, such as voltage frequency scaling, clock gating, and power gating, become less effective and applicable as technical scales. For instance, Intel technology blueprint shows that from 32 nm and beyond, with each technology generation, voltage and frequency scaling is tightly constrained, and the corresponding power benefits become marginal. On the other hand, this is not the first time we face the power consumption issue. Indeed, power consumption has been a recurring challenge for electronic system design. As shown in Fig. 2.1, power efficiency was one of the main motivations behind the technology transition from vacuum tube to bipolar, and then to CMOS during the past several decades. Device innovations have been the most effective solution. In addition, the historical trend implies that we will soon face another technology transition. However, CMOS technology has entered the nanometer regime. Further technology scaling becomes increasingly challenging. Emerging nano devices, such as carbon nanotubes, nanowires, and graphene, have demonstrated potential, but still face major challenges for large-scale chip and system integration. To build future

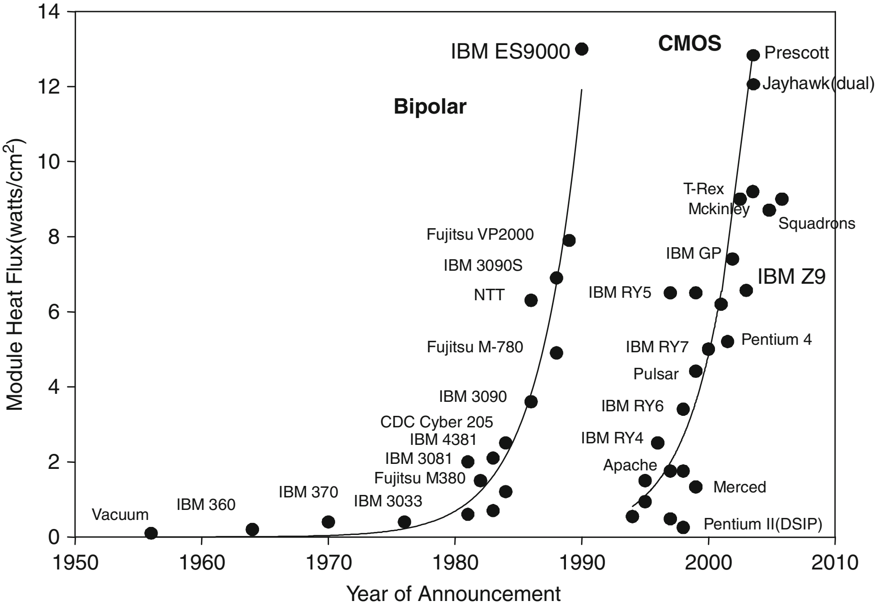


Fig. 2.1 Power consumption: a recurrent challenge

power-efficient systems, we need systematic innovations from system integration to device and fabrication technologies.

The rest of this chapter covers the basics of the IC power consumption issue. It first investigates the sources of IC power dissipation, and then discusses recent techniques for IC power modeling and analysis. Finally, it studies recently proposed power optimization techniques from circuit and physical design to system synthesis.

2.2 Source of Power Dissipation in CMOS Digital Circuits

In CMOS digital circuits, there are basically two sources of power dissipation, dynamic power dissipation and static power dissipation. Dynamic power dissipation arises from the transitions of the CMOS gate. It includes switching power dissipation and short-circuit power dissipation. When the logic level of the logic gate switches between ‘0’ (low voltage) and ‘1’ (high voltage), the parasitic capacitances are charged and discharged, during which the energy is converted into heat dissipation when current flows through the transistor channel resistance. This type of power dissipation is called switching power dissipation, which consumes most of the power used by CMOS circuits. Short-circuit power dissipation happens during the short transition interval when the output of a gate is changing in response to the changes of inputs. During the transition, the n-subnetwork and p-subnetwork of a CMOS gate both conduct simultaneously and a current flow from voltage source directly to ground is incurred. Static power consumption in CMOS circuit is mainly due to leakage. When gates are not transitioning, and because they are not fully turned off, there is a static leaking current flowing through the transistors, causing static power dissipation. Leakage power dissipation also contributes an important portion of the total power dissipation. The total power dissipation of a circuit is the sum of the three power sources. We will introduce each of them in detail in the following subsections.

2.2.1 Dynamic Power Dissipation

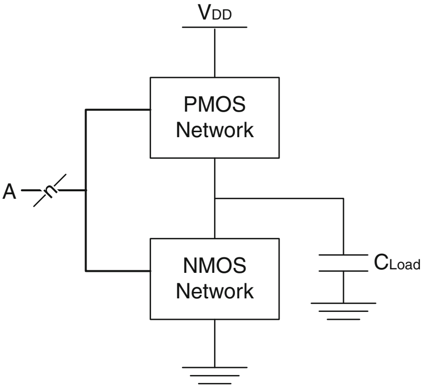
The dynamic power dissipation due to the switching of CMOS circuit, P_{swi}, can be calculated by the following equation:

$$P_{\mathrm{swi}} = \alpha C_{\mathrm{load}} V_{\mathrm{dd}} V_{\mathrm{out}} f \tag{2.1}$$

where α is the switching activity, i.e., the fraction of clock cycles in which the circuit switches, C_{load} is the load capacitance switched as shown in the Fig. 2.2, V_{dd} is the supply voltage, V_{out} is the output voltage, and f is the clock frequency. Usually V_{out} is equal to V_{dd} in the CMOS digital circuit. From this we have:

$$P_{\mathrm{swi}} = \alpha C_{\mathrm{load}} V_{\mathrm{dd}}^2 f \tag{2.2}$$

Fig. 2.2 CMOS gate with load capacitance



We can see from the equation that the larger the switched capacitance and the more frequently the circuit switches, the larger the dynamic power dissipation will be. Increasing V_{dd} will speed up the transistor transition, however, the switching power will increase quadratically with V_{dd}.

The energy consumed in the switching can be calculated based on the power consumption as:

$$\text{Energy} = \int p(t)\,\mathrm{d}t \tag{2.3}$$

where $p(t)$ is the power consumption of the circuit varying with time, which equals to the current from the source, I_{source}, times source voltage, i.e., $p(t) = I_{\mathrm{source}}(t)V_{\mathrm{dd}}$. Substitute $p(t)$ into Equation (2.3), and we obtain the expression for energy consumption as

$$\text{Energy} = \int I_{\mathrm{source}}(t)V_{\mathrm{dd}}\,\mathrm{d}t = \int C_{\mathrm{load}}\frac{\mathrm{d}_{\mathrm{out}}}{\mathrm{d}t}V_{\mathrm{dd}}\,\mathrm{d}t = C_{\mathrm{load}}V_{\mathrm{dd}}\int d_{\mathrm{out}} \tag{2.4}$$

When V_{out} is equal to V_{dd}, the energy drawn from the power supply is $C_{\mathrm{load}}V_{\mathrm{dd}}^2$.

Next, we will introduce how load capacitance is composed and how to obtain switching activities.

2.2.1.1 Components of Load Capacitance

First, we observed that in CMOS digital circuit, all gates drive other gates through on-chip interconnects. Figure 2.3a shows the capacitance in a MOSFET. We use Fig. 2.3b to illustrate the parasitic capacitance components that one gate usually drives. The overall load capacitance of the driver can be modeled as the parallel combination of the gate capacitances of the transistors that it drives C_{g}, the interconnect wire capacitance C_{int}, and its own drain-to-body capacitance C_{db}.

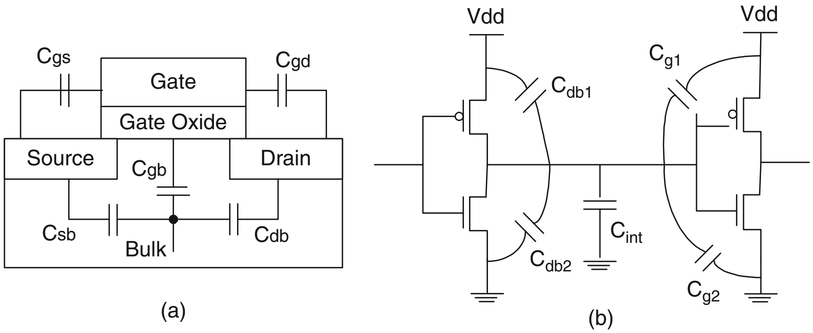


Fig. 2.3 **(a)** Capacitance in a MOSFET, **(b)** an inverter driving another showing parasitic load capacitance

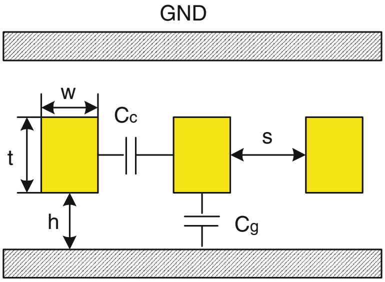


Fig. 2.4 Interconnect capacitance

Interconnect capacitance mainly comprises the area and fringe capacitance to the plane C_g and the coupling capacitance between neighboring wire C_c. Figure 2.4 shows the structure of the interconnect capacitance. The combined area and fringing-field capacitance C_g is given by [2]

$$C_g = \varepsilon_{ox} \left[\frac{w}{h} - \frac{t}{2h} + \frac{2\pi}{\ln\left(1 + \frac{2h}{t} 1 + \sqrt{1 + \frac{t}{h}}\right)} \right] \times l \tag{2.5}$$

$$w \geq \frac{t}{2} \tag{2.6}$$

where ε is the dielectric of the oxide insulation. l is the length of the metal wire. w, t, h, and s are the width and height of the wire, the distance of the wire to the plane, and the wire spacing as shown in the figure. The mutual-coupling capacitance between two wires can be calculated with different complexity [3].

We can see from the equation that the interconnect capacitance is proportional to the ratios between w, h, t, and s, which is determined by the design rule of the corresponding technology node. As the technology node continues scaling, there is a corresponding shrinking of the parameters. While the reduction of ε, w, and t

helps to reduce the total capacitance, C_g will increase with the decrease of h, and C_c increases with the decrease of s. As a result, the total interconnect capacitance first decreases with technology node scaling and then increases [4].

2.2.1.2 Switching Activity

Dynamic power consumption depends on the switching activity of the signals involved. In this context, the switching activity of the signals, α, can be defined as the average number of 0–1 transitions per clock cycle. Since there are equal probabilities to change from 0 to 1 and from 1 to 0, α is also equal to half of the total transitions of the node per cycle. If, in a time period with N clock cycles, the total number of switches for a signal is $n(N)$, then the switching activity of the signal can be calculated by $\alpha = \frac{n(N)}{2N}$. When gate outputs have glitches, they also contribute to the switching activities. Glitches are defined as the uncontrolled appearances of signal transitions. In some cases, it is found that the power consumption caused from glitches can account for up to 67% of the total dynamic power [5].

2.2.1.3 Clock Power Dissipation

Fully synchronous operation with clock signals has been the dominant design approach for digital systems. Recently, as technology nodes scale down to submicron regime, the clock frequencies of CMOS digital systems approach gigahertz range. Carrying large loads and switching at high frequency, clock power dissipation is a major source of the overall dynamic power dissipation in a digital system.

The dynamic power dissipated by switching the clock follows:

$$P_\mathrm{clk} = fV_\mathrm{dd}^2(C_\mathrm{l} + C_\mathrm{d}) \tag{2.7}$$

where C_l is the total load on the clock and C_d is the capacitance in the clock driver.

For example, for the H-tree-based global clock routing commonly used in digital circuits, as shown in Fig. 2.5, the source-to-sink delay is balanced to reduce the

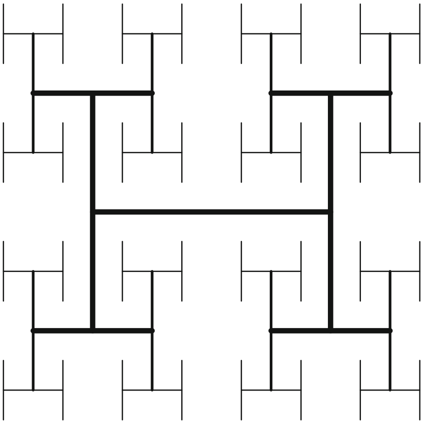


Fig. 2.5 H-tree clock distribution network

clock skew at the terminal. Given a total number of clock terminals N, the input capacitance at the terminal C_{in}, the unit length wire capacitance C_{wire}, the chip dimension D, and level of the H-tree h, its C_{l} can be estimated as [6]:

$$C_{\text{l}} = NC_{\text{in}} + 1.5(2^h - 1)DC_{\text{wire}} + \alpha\sqrt{N4^h}C_{\text{wire}} \tag{2.8}$$

It can be seen that the clock load capacitance, hence clock power dissipation, increases as the number of clocked terminals and the chip dimensions increase.

To ensure fast clock transitions, two common clock driving schemes, single driver or distributed buffer, are used to drive the large load capacitance on a clock. The single driver scheme uses a large buffer at the clock source. In this scheme, wire sizing is used to reduce the clock delay. Branches closer to the clock source are made wider. It also helps to reduce clock skew caused by asymmetric clock tree loads and wire width deviations [7, 8]. In the distributed buffer scheme, intermediate buffers are inserted in various parts of the clock tree. Relatively small buffers can be flexibly placed across the chip to save area and reduce clock delay. Figure 2.6 illustrates the two driving schemes. In any scheme, C_{l} and C_{d} need to be calculated for each buffer and summed to obtain the total capacitance for clock power dissipation.

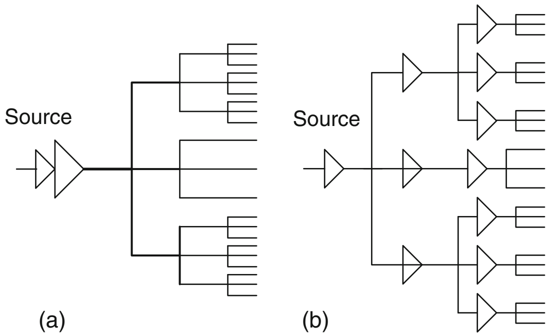


Fig. 2.6 Two clock tree driving schemes: **(a)** single driver scheme, **(b)** distributed buffers scheme

Traditionally, the load capacitance was largely contributed to by the capacitance of clock terminals. However, as technology advances into the deep submicron, device size is shrinking, while chip dimensions are increasing. This makes the interconnect capacitance dominant [6]. Reducing the interconnect capacitance may significantly reduce the overall power consumption. However, minimizing clock power consumption has to be considered together with meeting the constraints of clock skew and clock delay. Comparing the two clock schemes, the scheme of distributed buffers is preferred for low power design since it reduces both path length and load capacitance. Hence, it also reduces clock skew and minimizes the wire width. Thus, the total wire capacitance is kept at a minimum.

2.2.1.4 Short-Circuit Power Dissipation

The short-circuit power dissipation can be modeled using the following equation [9].

$$P_{\mathrm{sc}} = \frac{\beta}{12}(V_{\mathrm{dd}} - V_{\mathrm{tn}} - V_{\mathrm{tp}})^3 \frac{3\tau}{T} \tag{2.9}$$

where $V_{\mathrm{tn}}, V_{\mathrm{tp}}$ is the threshold voltage of the n and p transistor, respectively. β is the parameter determined by the rising and falling time of the output [9]. τ is the rise or fall time of the input signal, and T is the clock cycle of the input signal. However, since the gate may not switch during every clock cycle, the node activity factor must be added in. $\frac{1}{T}$ is revised to be $(\alpha_{10} + \alpha_{01})f$, where f is the input frequency [10]. On the other hand, later studies also show that P_{sc} closely varies with different load capacitances, and that the simple equation is mainly for slow input signals (large τ). Hence, various complicated models are derived to give a more accurate estimation of P_{sc} when needed [11, 12]. Since P_{sc} is normally less than 10% of the dynamic power, in most cases, it is neglected [10].

2.2.2 Leakage Power Dissipation

As a result of continued IC process scaling, which reduces transistor threshold voltage, channel length, and gate oxide thickness, the importance of leakage power consumption is increasing [12]. Presently, leakage accounts for 40% of the power consumption of modern 65 nm high-performance microprocessors, and will continue to increase as technology scales further [13]. Without leakage reduction techniques, this ratio will increase with further technology scaling. Indeed, the primary goal of high-k and metal gate research and development effort is to address the fast increasing MOS transistor gate leakage power dissipation [14]. The impact of circuit leakage effect on IC performance, power consumption, temperature, and reliability is fast growing. IC leakage power consumption, which was largely ignored in the past, has become a primary concern in low-power IC design. Leakage power must now be carefully considered and optimized during the entire IC design flow.

IC leakage current consists of various components, including subthreshold leakage, gate leakage, reverse-biased junction leakage, punch-through leakage, and gate-induced drain leakage [15], as shown in Fig. 2.7. Among these, subthreshold leakage and gate leakage are currently dominant, and are likely to remain dominant in the near future [1]. They will be the focus of our analysis.

Considering weak inversion drain-induced barrier lowering and body effect, the subthreshold leakage current of a MOS device can be modeled as follows [16]:

$$I_{\mathrm{subthreshold}} = A_{\mathrm{s}} \frac{W}{L} v_{\mathrm{T}}{}^2 \left(1 - \mathrm{e}^{\frac{-V_{\mathrm{DS}}}{v_{\mathrm{T}}}}\right) \mathrm{e}^{\frac{(V_{\mathrm{GS}} - V_{\mathrm{th}})}{n v_{\mathrm{T}}}} \tag{2.10}$$

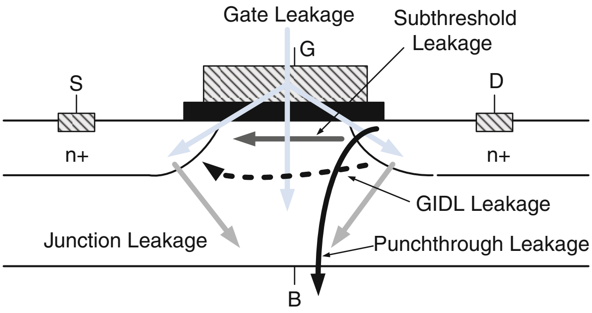


Fig. 2.7 Leakages current components in a MOS transistor

- where A_s is a technology-dependent constant,
- V_{th} is the threshold voltage,
- L and W are the device effective channel length and width,
- V_{GS} is the gate-to-source voltage,
- n is the subthreshold swing coefficient for the transistor,
- V_{DS} is the drain-to-source voltage, and
- v_T is the thermal voltage.

$V_{DS} \gg v_T$ and $v_T = \frac{kT}{q}$. Therefore, Equation (2.10) can be reduced to

$$I_{\text{subthreshold}} = A_s \frac{W}{L} \left(\frac{kT}{q} \right)^2 \mathrm{e}^{\frac{q(V_{GS}-V_{th})}{nkT}} \tag{2.11}$$

Equation (2.11) shows that, IC subthreshold leakage current has exponential dependency on circuit threshold voltage. As described in Section 2.4.5, through technology scaling, stringent circuit performance requirements impose continuous reduction of IC threshold voltage, which results in significant increase of IC leakage current. As a result, IC leakage power consumption has become a first-order design issue, especially in the mobile application sector. Furthermore, IC subthreshold leakage current is a strong function of temperature. As temperature increases, circuit subthreshold leakage current increases superlinearly.

Leakage of a MOS device results from tunneling between the gate terminal and the other three terminals (source, drain, and body). Gate leakage can be modeled as follows [17]:

$$I_{\text{gate}} = WLA_J \left(\frac{T_{\text{oxr}}}{T_{\text{ox}}} \right)^{\text{nt}} \frac{V_g V_{\text{aux}}}{T_{\text{ox}}^2} e^{-BT_{\text{ox}}(a-b|V_{\text{ox}}|)(1+c|V_{\text{ox}}|)} \tag{2.12}$$

- where A_J, B, a, b, and c are technology-dependent constants,
- nt is a fitting parameter with a default value of one,

- V_{ox} is the voltage across gate dielectric,
- T_{ox} is gate dielectric thickness,
- T_{oxr} is the reference oxide thickness,
- V_{aux} is an auxiliary function that approximates the density of tunneling carriers and available states, and
- V_g is the gate voltage.

Equation (2.12) shows that IC gate leakage current has strong dependency on device gate dielectric thickness. Through technology scaling, IC supply voltage reduces continuously in order to minimize circuit dynamic power consumption. However, to maintain good circuit performance, the thickness of transistor gate dielectric layer needs to reduce accordingly. Thus, IC gate leakage current increases exponentially. Modern transistor gate dielectric layer is only a few atoms thick. Gate leakage hence becomes a serious concern. To address this problem, intensive research and development efforts have been devoted searching for new gate dielectric material, i.e., high-k material, and the corresponding fabrication technology. Commercially solutions, such as metal-gate from Intel, have been widely deployed and demonstrated effective gate leakage reduction.

2.2.3 Limits of CMOS Circuits

Since the first transistor was fabricated in 1940s, the number of transistors per chip has continued to double every 18–24 months. This trend was observed by Gordon Moore of Intel, and is known as Moore's law. Accompanied by the growth in transistor density is the increase in reliability and decline of energy consumption per transistor transition. However, with continuous scaling, CMOS technology is approaching its scaling limits. Great attention is being paid to the limits of continued scaling [18–23]. Reference [20] defines a hierarchy of limits that have five levels: fundamental, material, device, circuit, and systems. Hu considers the reliability constraints on the scaling of MOS devices [19]. Currently, with CMOS scaling to below 32 nm, the primary obstacle arises from the static power dissipation, which is caused by leakage currents due to quantum tunneling and thermal excitations [22]. Next, we will discuss this power issue and the methods to overcome it.

2.2.3.1 Power-Constraint Scaling

Dynamic power can be adjusted to a limited extent by adjusting the load capacitance, the supply voltage, or the operational frequency. However, leakages, including subthreshold and gate-dielectric leakages, have unfortunately become the dominant barrier to further CMOS scaling, even for highly leakage-tolerant applications such as microprocessors.

As the channel length of a field effect transistor (FET) is reduced, the drain potential begins to strongly influence the channel potential, leading to drain-induced barrier lowering (DIBL). DIBL eventually allows electron flow between the source and the drain, even if the gate-to-source voltage is lower than the threshold voltage, leading to an inability to shut off the channel current with the gate. The channel current that flows under these conditions is called the subthreshold current. Subthreshold current contributes the most significant part of the static leakage consumption.

This short-channel effect (SCE) can be mitigated by reducing the gate oxide (to increase the control of the gate on the channel) and the use of thin depletion depth below the channel to the substrate, to shield the channel from the drain [24]. However, the reduction of gate oxide results in an increase of gate leakage current at the same time. At 90 nm CMOS, the power from gate leakage is comparable to the power used for switching of the circuit [24]. Thus, further reduction of gate oxide thickness would lead to unreasonable power increases. Alternatively, further decrease of the depletion region degrades gate control on the channel and slows down the turn on speed of the FET. For bulk CMOS, increased body doping concentration could be also employed to reduce DIBL; however, at some point it would also increase the subthreshold swing. Therefore, a higher threshold voltage is needed to keep the subthreshold current adequately low. Similarly, decreasing the body doping concentration could improve the subthreshold swing, but could degrade DIBL. Hence it is difficult to reduce both DIBL and subthreshold current for the bulk-silicon device design [24].

Besides gate tunneling leakage, other sources of tunneling leakage current are band-to-band tunneling between the body and drain of an FET and direct source-to-drain tunneling through the channel barrier [22].

In order to mitigate the impact of short channel effect to FET scaling, and reduce leakage power dissipation, new FET structure or even new technologies can be considered, such as carbon nanotube or graphene-based FETs.

2.2.3.2 Effects of Variations

Due to the limited resolution of photolithographic process, there is random process variation on the transistor dimensions from wafer to wafer and die to die around 10–20% [25], which increases with technology scaling. As discussed before, among the multiple leakage sources, subthreshold leakage and gate leakage play the most important role. According to the Equations (1.10), (1.11), (1.12), we can see that leakage is determined by the device dimensions (feature size, oxide thickness, junction depth, etc.), doping profiles, and temperature. Hence, as a result, statistical variation in each of the device parameters causes a large variation in each of the leakage components. For example, for subthreshold leakage, it linearly depends on the dimensions of the device and exponentially depends on the gate threshold voltage. Voltage on the other hand depends on oxide thickness, implant impurity, surface charge, etc. For gate leakage, it mainly depends on oxide thickness too exponentially. Hence the variation of gate oxide thickness will result in a large variation of leakage. To estimate the leakage current, we need to find its mean and standard variation using statistics. It can be done through either analysis or simulation. Reference [25] generalizes the statistical analysis procedure for estimating the mean *mu* and the standard deviation σ of a leakage component considering variation in a single parameter (say x) as the following:

- Express the current as function of the variable x as $g(x)$.
- Estimate mean of $g(x) : \mu[g(x)] = g(\mu_x + \frac{g^2(\mu_x)}{2}\sigma_x^2$.
- Estimate mean of $g(x)^2 : \mu[g(x)^2] = g^2(\mu_x) + \frac{(g^2)^2(\mu_x)}{2}\sigma_x^2$.
- Estimate standard deviation of $g(x) : \sigma[g(x)] = \sqrt{\mu[g(x)^2] - [\mu[g(x)]^2}$.

To derive the estimation of leakage component under variation of multiple parameters, according to the function of the variables, one can assume the relationship between variables to be independent or correlated, then apply the statistics equations for mean and standard deviation computation accordingly. The estimation can also be performed using Monte Carlo simulation method [26]. Using simulation method, the full chip level leakage estimation can be enabled with or without consideration of parameter correlation [27, 28].

Process variation degrades parametric yield by impacting both power consumption and performance of a design. This problem is enlarged by the fact that circuit timing is inversely proportional to power. For example, a reduction in channel length results in improved performance but also causes an exponential increase in leakage power. This inverse correlation makes it challenging to meet both power and frequency constraints. Many manufactured chips that meet timing end up exceeding power budget while other chips within the power limit fail to deliver the required performance [29]. Traditional parametric yield analysis of high-performance integrated circuits is mainly based on the frequency (or delay). For current design with power consumption as an important factor, integrated approachs to accurately estimate and optimize the yield when both a frequency and power limits are imposed on a design are proposed [29, 30].

2.3 Power Estimation

When designing VLSI circuits, the designers need to accurately estimate the silicon area, the expected performance, and power dissipation before fabricating the chip. Power estimation can be performed at different level resulting in tradeoff between estimation accuracy and efficiency. There are typically two types of power estimation: average power and peak power. Average power determines battery life while maximum or peak power is related to circuit cooling cost and reliability.

2.3.1 Dynamic Power Estimation

Since dynamic power consumption contributes a large portion of the total power consumption, dynamic power estimation is critical to the low-power design. Great amount of work has been conducted on dynamic power estimation on different design levels using simulation-based or analytic methods. It has been observed that when estimation is performed based on function blocks, the algorithm can produce result in a short time. However, when signal switch activities and internal load capacitances are required on individual lines for more accurate estimation, the results generating rate can be very slow. Next, we will introduce the commonly used estimation method for dynamic power estimation.

2.3.1.1 Simulation-Based Estimation

Circuit simulation-based techniques [31] simulate the circuit with a representative set of input vectors. This method is accurate and generous to different design. However, it suffers from the memory and execution constraint for large scale design. In general, it is difficult to generate a compact vector set, if it is not possible to go through all the input vectors, to calculate accurate activity factors at the circuit nodes.

To alleviate this problem, A Monte Carlo simulation approach for power estimation is proposed in [32]. This approach assumes randomly generated input patterns at the circuit inputs and simulate the power dissipation per time interval T. Based on the assumption that power dissipation over any interval has a normal distribution, given a error percentage and confidence level, the average power consumption is estimated. Note that the normality assumption may not hold in some cases, then the approach will converge to a wrong estimation.

2.3.1.2 Probabilistic Power Estimation

To estimate the dynamic power consumption, one has to calculate the switching activity of each internal node of the circuit. Assuming that the circuit inputs are independent to each other, probabilistic method can be used to estimate the switching activities of node n based on its signal probability prob(n). Then the switching activity of n is the probability of the signal switching from 0 to 1, which equals to $\alpha_n = \text{prob}(n)(1 - \text{prob}(n))$. If a network consists of simple gates and has no reconvergent fanout nodes, i.e., tree-like structure, then the exact signal probabilities can be computed during a single traversal of the network using the following equations [33]:

$$\begin{aligned} &\text{not gate} : \text{prob(out)} = 1 - \text{prob(in)} \\ &\text{and gate} : \text{prob(out)} = \prod_{i \in \text{inputs}} \text{prob}(i) \\ &\text{or gate} : 1 - \prod_{i \in \text{inputs}} (1 - \text{prob}(i)) \end{aligned} \tag{2.13}$$

For network with reconvergent fanout, this simple algorithm yields approximate results for signal probabilities.

Another way for signal probability for general circuits is based on Shanon's expansion and BDD graph [34, 35]. According to Shanon's expansion,

$$f = x_i f_{x_i} + x_i' f_{x_i'} \tag{2.14}$$

where f_{x_i} means the function value when $x_i = 1$, i.e., $f(x_1, \cdots, x_{i-1}, 1, x_{i+1}, \cdots, x_n)$, and $f_{x_i'}$ correspondingly means the function value when substitute x with 0. Noted that $x_i, f_{x_i}, x_i', f_{x_i'}$ are independent with each other, the signal probability of f can be calculate as:

$$\begin{aligned} \alpha_f &= \text{prob}(x_i f_{x_i} + x_i' f_{x_i'}) \\ &= \text{prob}(x_i f_{x_i}) + \text{prob}(x_i' f_{x_i'}) \\ &= \text{prob}(x_i)\,\text{prob}(f_{x_i}) + \text{prob}(x_i')\,\text{prob}(f_{x_i'}) \end{aligned} \tag{2.15}$$

Boolean difference is another similar method for probabilistic switching activity estimation in combinational logic described in detail in [36]. Note that all the above estimation is based on zero-delay model. Estimation under a real delay model using symbolic simulation is presented in [37]. Above discussion mainly focuses on estimation for combinational logic. The estimation method for sequential circuit can greatly differ from the combinational ones. The average switching activity estimation for finite state machines (FSM) need to consider two impacts: (1) The probability of the circuit being in each of its possible states. (2) The probability of present state line inputs. The work in [38] presents the method to compute the exact state probabilities of the FSM using Chapman-Kolmogorov equations. For each state S_i in total K states, I_{ij} specify the input combination which transfers the FSM from state i to state j. Given static probabilities for the primary inputs to the machine, we can compute the conditional probability of going from S_i to S_j, prob$(S_j|S_i)$. For each state S_j, we can write the equation:

$$\text{prob}(S_j) = \sum_{S_i \in \text{instates}(S_j)} \text{prob}(S_i)\text{prob}(S_j|S_i) \tag{2.16}$$

where instate(S_i) is the set of fanin states of S_i in the state transfer graph. Given K states, we obtain K equations. Finally, we have a last equation:

$$\sum_j \text{prob}S_j = 1 \tag{2.17}$$

Different probabilities of the states can be obtained by solving the set of linear equations. We use an example in Fig. 2.8 to illustrate the procedure. Suppose all the FSM inputs are with signal probability 0.5. We can get the set of equations as:

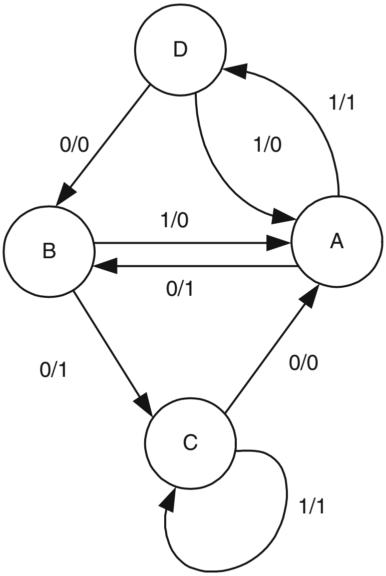


Fig. 2.8 State transfer graph for an example FSM

$$
\begin{aligned}
\text{prob}(D) &= 0.5 \times \text{prob}(A) \\
\text{prob}(A) &= 0.5 \times \text{prob}(D) + 0.5 \times \text{prob}(B) + 0.5 \times \text{prob}(C) \\
\text{prob}(B) &= 0.5 \times \text{prob}(D) + 0.5 \times \text{prob}(A) \\
\text{prob}(A) &+ \text{prob}(B) + \text{prob}(C) + \text{prob}(D) = 1
\end{aligned}
\tag{2.18}
$$

By solving the set of equations, we can get $\text{prob}(D) = \frac{1}{6}, \text{prob}(A) = \frac{1}{3}$, $\text{prob}(B) = \frac{1}{7}$, and $\text{prob}(C) = \frac{1}{8}$.

2.3.1.3 Power Macromodeling

In high-level power estimation, the circuit is described as a set of blocks with known internal structure. The power dissipation of each block is estimated using a macromodel [39–41]. Here macromodel means some form of equation which can best match the simulated power dissipation numbers. The equation can only contain the variable of primary inputs, or the statistics can be used to improve the macromodel. For example, the statistics based macromodel can be:

$$\text{Power} = f(\text{Mean}_A, \text{Mean}_B, \text{SD}_A, \text{SDB}, \text{TC}_A, \text{TC}_B, \text{SC}_{AB}, gl_A, gl_B) \tag{2.19}$$

where A and B are the inputs to the block. Mean, SD, TC, SC, and gl indicate mean, standard deviation, time correlation, spatial correlation, and glitching factor, respectively. The macromodel can be built based on analytic derivation or simulation. A regression method for constructing macromodels are illustrated in Fig. 2.9 [41].

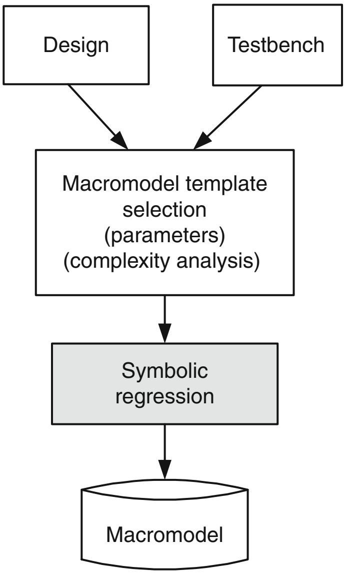

Design
Testbench
Macromodel template selection (parameters) (complexity analysis)
Symbolic regression
Macromodel

Fig. 2.9 Power macromodel generation using regression method

2.3.2 Leakage Power Estimation

Researchers have developed a variety of techniques to characterize IC leakage power consumption, ranging from architectural level to device level [15, 42–46]. We now survey leakage analysis work spanning these design levels.

Device-level leakage power estimation generally relies on models for individual transistor leakage mechanisms. Transistor leakage power consumption is a function of device physical properties and fabrication process. For bulk CMOS, the main control variables for leakage are device dimensions (feature size, oxide thickness, junction depth, etc.) and doping profiles in transistors [15]. Based on those physical characteristics, leakage models can be developed to predict the components of leakage, e.g., subthreshold leakage, gate leakage, and junction leakage [47]. Generally, technology constants provided by foundry can be used in such models. Transistor-level simulators incorporating these models can accurately predict leakage [48]; however they are computationally expensive due to iteratively solving complex leakage formulas.

In addition to its dependence on device parameters, IC leakage power consumption is affected by a number of circuit level parameters, e.g., the distribution of device types (NMOS and PMOS), geometries (channel width and length), and control voltages. Numerous circuit-level leakage estimation techniques have been proposed. Sirichotiyakul et al. presented an accurate and efficient average leakage calculation method for dual-threshold CMOS circuits that is based on graph reduction techniques and simplified nonlinear simulation [49]. Lee et al. proposed fast and accurate state-dependent leakage estimation heuristics using circuit block level look-up tables, targeting both subthreshold and gate leakage [50]. To conduct full-chip leakage estimation accurately, it is possible to model and sum the leakage currents of all gates [51, 52]. However, this is too computationally intensive for use in earlier design stages of very-large scale integration circuits.

For architectural leakage models [42], design parameters characterizing microarchitectural design styles and transistor sizing strategies can be extracted from typical logic and memory circuits. Do et al. proposed high-level dynamic and leakage power models to accurately estimate physically partitioned and power-gated SRAM arrays [53]. Given a set of inputs, Gopalakrishnan et al. used a bit-slice cell library to estimate the total leakage energy dissipated in a given VHDL structural datapath [54]. Kumar et al. presented a state-dependent analytical leakage power model for FPGAs [55]. The techniques described in this paragraph provide reasonable accuracy for early design stage leakage estimation, as long as temperature is fixed. However, they do not consider temperature variations.

IC leakage power consumption is a strong function of temperature, e.g., subthreshold leakage increases super-linearly with chip temperature. In modern microprocessors, power density has reached the level in a nuclear reactor core, causing high chip temperatures and hence high leakage power consumptions. Due to time-varying workload and operating states with different power consumption levels (up to 25 power states in the Core$^{\text{TM}}$ Duo processor [56]) and uneven on-chip power density distribution, large on-chip temperature variations and gradient are common in high-performance ICs. For example, ICs may have larger than 40°C temperature difference [57], causing high on-chip leakage variation. In summary, increasing chip temperature and on-chip temperature variation significantly affect IC leakage power consumption [58]. Therefore, accurate leakage power analysis requires temperature to be considered.

Some researchers have developed temperature-dependent architectural leakage power models. Zhang et al. developed HotLeakage, a temperature-dependent cache leakage power model [59]. Su et al. proposed a full-chip leakage modeling technique that characterizes the impact of temperature and supply voltage fluctuations [60]. Liao et al. presented a temperature-dependent microarchitectural power model [61].

Figure 2.10 shows a typical temperature-aware power estimation flow. Power consumption, including dynamic power and leakage power, is initially estimated at a reference temperature. Given an IC design, initial dynamic power for each circuit macro block is determined by estimated macro block workloads, switching factors, and supply voltage using commercial tools such as Synopsys Power Compiler and PrimePower in its Galaxy Design Platform [62]. Initial leakage power can be obtained by HSPICE simulation or other efficient high-level methods [49, 51, 52]. The estimated power profile is then provided to a chip-package thermal analysis tool to estimate circuit thermal profile. This thermal profile is, in turn, used to update circuit macro block leakage power. This iterative process continues until leakage power consumption converges.

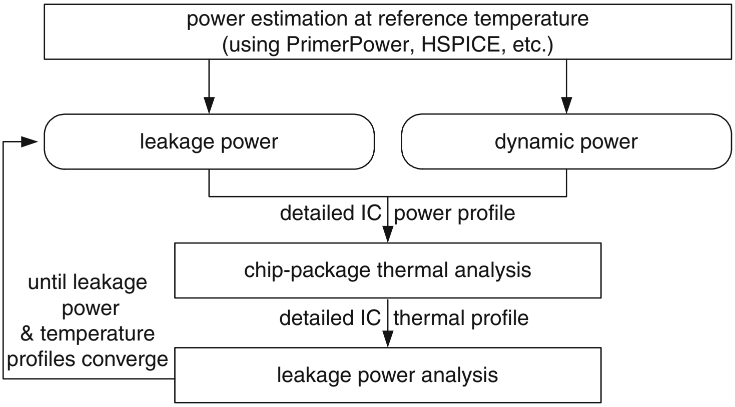

power estimation at reference temperature
(using PrimerPower, HSPICE, etc.)
leakage power
dynamic power
detailed IC power profile
chip-package thermal analysis
detailed IC thermal profile
leakage power analysis
until leakage power & temperature profiles converge

Fig. 2.10 Temperature-aware leakage power analysis flow

2.3.2.1 Thermal Analysis

IC thermal analysis is the simulation of heat transfer through heterogeneous material among heat producers (e.g., transistors) and heat consumers (e.g., heat sinks attached to IC packages). Modeling thermal conduction is analogous to modeling electrical conduction, with thermal conductivity corresponding to electrical conductivity, power dissipation corresponding to electrical current, heat capacity corresponding to electrical capacitance, and temperature corresponding to voltage.

The equation governing heat diffusion via thermal conduction in an IC follows.

$$\rho c \frac{\partial T(\mathbf{r},t)}{\partial t} = \triangledown(k(\mathbf{r}) \triangledown T(\mathbf{r},t)) + p(\mathbf{r},t) \tag{2.20}$$

subject to the boundary condition

$$k(\mathbf{r},t)\frac{\partial T(\mathbf{r},t)}{\partial n_i} + h_i T(\mathbf{r},t) = f_i(\mathbf{r},t) \tag{2.21}$$

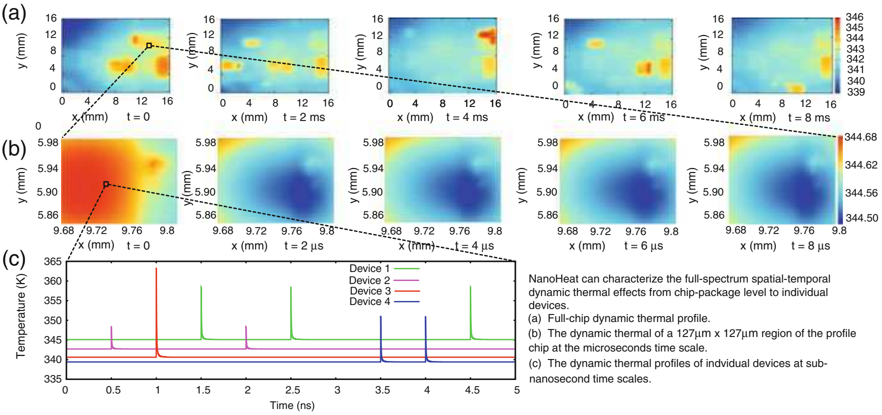
In Equation (2.20), ρ is the material density, c is the mass heat capacity, $T(\mathbf{r},t)$ and $k(\mathbf{r})$ are the temperature and thermal conductivity of the material at position $\mathbf{r}$ and time t, and $p(\mathbf{r},t)$ is the power density of the heat source. In Equation (2.21), n_i is the outward direction normal to the boundary surface i, h_i is the heat transfer coefficient, and f_i is an arbitrary function at the surface i. Note that, in reality, the thermal conductivity, k, also depends on temperature. Recently proposed thermal analysis solutions begin to support arbitrary heterogeneous three-dimensional thermal conduction models. For example, a model may be composed of a heat sink in a forced-air ambient environment, heat spreader, bulk silicon, active layer, and packaging material or any other geometry and combination of materials.

In order to do numerical thermal analysis, a seven point finite difference discretization method can be applied to the left and right side of Equation (2.20), i.e., the IC thermal behavior may be modeled by decomposing it into numerous rectangular parallelepipeds, which may be of non-uniform sizes and shapes. Adjacent elements interact via heat diffusion. Each element has a power dissipation, temperature, thermal capacitance, and a thermal resistance to adjacent elements. For an IC chip-package design with N discretized elements, the thermal analysis problem can then be described as follows.

$$\mathbf{C}T(t)' + \mathbf{A}T(t) = Pu(t) \tag{2.22}$$

where the thermal capacitance matrix, $\mathbf{C}$, is an $[N \times N]$ diagonal matrix; the thermal conductivity matrix, $\mathbf{A}$, is an $[N \times N]$ sparse matrix; $T(t)$ and $P(t)$ are $[N \times 1]$ temperature and power vectors; and $u(t)$ is the time step function. For steady-state analysis, the left term in Equation (2.22) expressing temperature variation as function of time, t, is dropped. For either the dynamic or steady-state version of the problem, although direct solutions are theoretically possible, the computational expense is too high for use on high-resolution thermal models.

Both steady-state and dynamic analysis methods have been developed in the past for full-chip IC thermal analysis [60, 63, 64, 65–69].These solutions, however, are unable to support nanometer-scale device-level spatial and temporal modeling granularities. Furthermore, they rely on the Fourier thermal physics model. The Fourier model with fixed material thermal conductivities fails at length scales comparable to the phonon mean free path (the average distance phonons travel before suffering scattering events), and at time scales comparable to that on which phonon scattering events occur [70]. Current device sizes and switching speeds have already reached those limits. This yields the Fourier equation inadequate for modeling device-level thermal effects. Recently, Allec et al. [71] proposed a multi-scale IC thermal analysis solution that can characterize the device-level thermal effect, producing IC thermal profiles with transistor-level spatial resolution. Figure 2.11 shows the runtime thermal profile of an IC design. It shows that, from chip–package level to



NanoHeat can characterize the full-spectrum spatial-temporal dynamic thermal effects from chip-package level to individual devices.
(a) Full-chip dynamic thermal profile.
(b) The dynamic thermal of a 127µm x 127µm region of the profile chip at the microseconds time scale.
(c) The dynamic thermal profiles of indvidual devices at sub-nanosecond time scales.

Fig. 2.11 Thermal characterization of nanometer-scale ICs

individual devices, the IC thermal characteristic is heterogenenous and dynamic, which has direct impact on IC performance, power consumption, short-time scale, and lifetime reliability.

2.4 Power Optimization and Management

In order to meet the power requirement, lengthen the battery life, and increase the chip reliability, power optimization has become a crucial design aspect. Power optimization can also be performed at different design phases, targeting dynamic power reduction or leakage power reduction. The higher the level power optimization is performed, the more power reduction that can be obtained. In the following subsections, we will discuss the optimization procedures for low power dissipation at algorithm, architecture, gate, and circuit levels.

2.4.1 High-Level Synthesis for Low Power

As we have discussed in Section 2.2.1, the dynamic power dissipation is proportional to the value of switching activity, load capacitance, and source voltage. Hence, in low power design, we will reduce the power dissipation of circuits by reducing each of the factors first in the high level synthesis. Due to the large freedom for synthesis from high-level specification to low-level implementation, there is great potential of producing a large improvement in power dissipation during the synthesis procedure. At the behavioral level, since operations have not been assigned, the execution time and hardware allocation have not been performed, a systematic design space exploration can be performed to search for a design point satisfying the given power dissipation, delay, and area constraints. Traditionally, behavioral synthesis has targeted optimization of hardware resource usage and average clock cycles for execution of a set of tasks. With the increase in concern for power consumption, a great amount of studies have been carried out to examine the efficacy of behavioral level techniques to reduce power dissipation. The essential methodology is to reduce the number of power-consuming operations, restructure the microarchitecture to reduce switching activities, and balance the microarchitecture to reduce glitches. Reference [72] proposes an iterative algorithm for performing scheduling, clock selection, module selection, and resource allocation and assignment simultaneously with an aim of reducing the power consumption in the synthesized data path. Next, we introduce some of these techniques in detail.

2.4.1.1 Transformation

Some parts of the CDFG can be restructured to reduce the number of modules used, and hence, reduce the power consumption needed to realize the same functionality. For example, in Fig. 2.12a, two multipliers will switch in one step. However, after

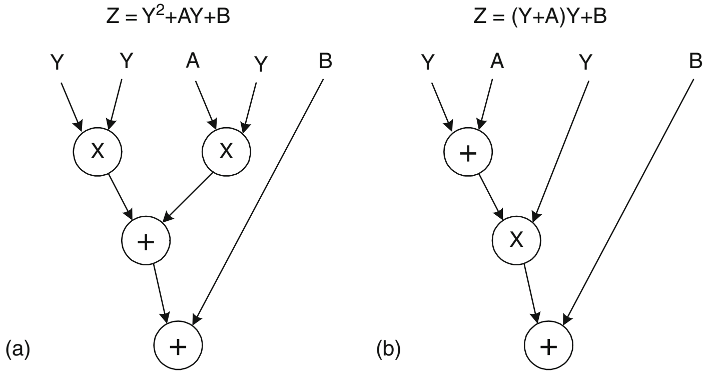


Fig. 2.12 Restructure of the circuit to reduce power consumption

a slight structural transformation, only one multiplier switches in one step as in Fig. 2.12b.

2.4.1.2 Scheduling

The behavioral level synthesis is based on the behavioral description of the circuit in the form of a control-dataflow graph (CDFG). A CDFG is a directed graph whose vertices consist of arithmetic, logical and comparison operations, delay operators, and special branch, merge, loop entry, and loop exit vertices that represent control flow constructs [72]. The CDFG contains data (control) flow edges that represent data (control) dependencies between operations. An example CDFG shown in Fig. 2.13a represents the computation of $z = |a - b|$. The process of scheduling assigns each operation in the CDFG to one or more cycles or control steps. Figure 2.13b, c illustrate two possible schedules of the task graph using minimum resources. The horizontal dotted lines labeled with numbers indicate the clock edges, i.e., the boundaries between control steps. The task takes three clock cycles to execute. In Fig. 2.13b, since two subtraction operations are preformed in different clock cycles, one subtractor is sufficient to implement the functions, i.e., executing subtraction 1(2) in clock 1(2). For the scheduling in Fig. 2.13c, although subtraction 1 and 2 are in one step, they are mutually exclusive. They will not execute at the same time; hence, they can be realized by one subtractor. Therefore, both schedules use one comparator, one MUX and one subtractor. However, the subtractor will be active in both clocks in the first scheduling, while in the second scheduling, the subtractor is only active during one clock and hence will consume less power.

2.4.1.3 Module Selection

Module selection refers to the process of selecting, for each operation in the CDFG, the type of functional unit that will perform the operation. In order to fully explore the design space, it is necessary to have a diverse library of functional units which

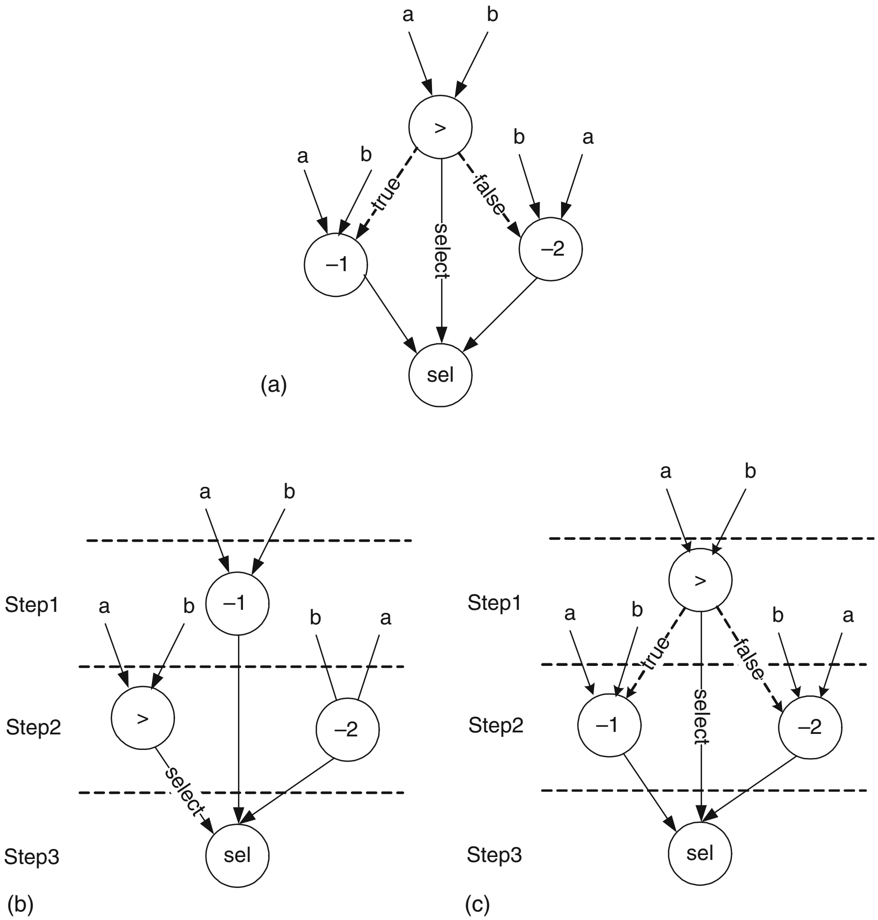


Fig. 2.13 Different schedules for the example circuit

perform the same function; however, do so with different logic implementation, and hence, different area, delay, and power consumption. For example, there can be many types of adders including ripple-carry-adder, carry-lookahead-adder, carry-select-adder, etc. A faster module is typically more expensive in terms of area and switched capacitance, i.e., power consumption. It is possible to perform tradeoffs in area, delay, and power using module selection. Suppose there are two types of multipliers, a WAL-MULT with 40 ns delay and an ARR-MULT with 60 ns delay. The power consumption of the ARR-MULT is much less than the WAL-MULT. When performing module selection as illustrated in Fig. 2.14, by replacing the WAL-M module with an ARR-M, the multiplication will take multiple clock cycles; however, the delay of the circuit will not be affected and power consumption is saved.

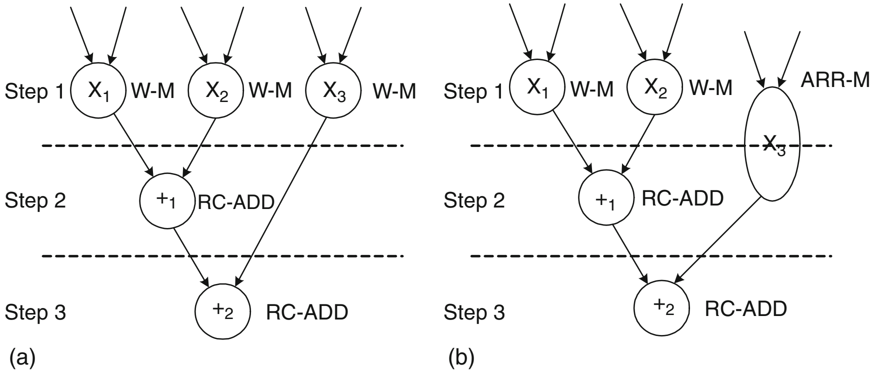


Fig. 2.14 Module selection for low power

2.4.1.4 Resource Sharing

Resource sharing refers to the use of the same hardware resource (functional unit or register) to perform different operations or store different variables. Resource sharing is performed in the step of hardware allocation and assignment, such as register binding, in the behavioral synthesis flow. These processes decide how many resources are needed and which type of resource to use to implement an operation or variable storage. Scheduling and clock selection, discussed later, affect how resource sharing is performed. In turn, resource sharing significantly affects both the physical capacitance and switching activity in the data path. Heavy resource sharing tends to reduce the number of resources used and the physical capacitance, but increases the average switching activity in the data path. Sparsely shared architectures have lower average switching activity, but higher physical capacitance and area usage. A detailed analysis of the effect of resource sharing on switched capacitance can be found in [72].

2.4.1.5 Clock Selection

Clock selection refers to the process of choosing a suitable clock period for the controller/data path circuit. Given the clock period, the execution time of the CDFG, which is equal to the input sample period, will be divided into several clock cycles. The choice of the clock period is known to have a significant effect on both area and performance [73]. However, it also has a great impact on power consumption as pointed out by [74]. A longer clock period is helpful for reducing the switching power consumption in the clock network, reducing the number of registers needed, and using modules with longer delay but less power consumption. However, on the other hand, a long clock period will contain more modules in its cycle, which tends to create more glitches between modules and inhibit the amount of resource sharing. Larger time slacks also tend to appear in long clock periods.

2.4.2 Architectural Level Power Optimization

After the circuit specification is synthesized into architecture-level functional units, several optimization methods aiming to reduce switching activity, source voltage or load capacitance can be performed based on the microarchitecture, and its performance, as discussed below.

2.4.2.1 Architecture-Driven Voltage Scaling

As seen from the equation for dynamic power dissipation, a large reduction in power dissipation can be obtained if the supply voltage is scaled down, as V_{dd} appears as a squared term in the expression. However, one direct side effect is the increase in circuit delay.

$$\mathrm{Delay} = \frac{kV_{\mathrm{dd}}}{(V_{\mathrm{dd}} - V_{\mathrm{t}})^{\alpha}} \tag{2.23}$$

where k and α are some parameters. One scenario to use voltage scaling is when there is time slack in the pipeline after behavioral synthesis. Another simple way to maintain throughput while reducing the supply voltage is to leverage parallelism or use a more deeply pipelined architecture. For example, the original circuit is as in Fig. 2.15a. If the combinational logic is further divided into N pipelines as in Fig. 2.15b, the voltage can be scaled down to close to N (equal to N, if V_{t} is zero), while throughput is not affected. However, note that by inserting more pipeline registers, total switching capacitance also increases. Finally, a less than N times power reduction can be achieved. With the price of area increasing, parallel architectures can reduce power dissipation even further. As illustrated in Fig. 2.15c, a parallel architecture duplicates the input registers and functional units to parallel process N samples, however, only one output register stores the corresponding result selected from the N units in one clock cycle. Since one unit only processes one sample during N clock cycles, the circuit delay for the unit can be N times the original circuit delay. By calculation, it is obtained that the source voltage can be scaled to $1/N$ of the original voltage. Then the power consumption of the parallel architecture becomes $\frac{1}{N^2}$ of the previous power consumption, as shown below.

$$\begin{aligned} P_{\mathrm{para}} &= NC_{\mathrm{load}}\frac{V_{\mathrm{dd}}^2}{N^2}\frac{f_{\mathrm{sample}}}{N} \\ &= \frac{1}{N^2}C_{\mathrm{load}}V_{\mathrm{dd}}^2 f_{\mathrm{sample}} \end{aligned} \tag{2.24}$$

2.4.2.2 Power Management with Clock Gating

On any given clock cycle, a part of the architecture may be idle. If so, it is possible to save substantial power by disabling clocks to the areas of the architecture that are

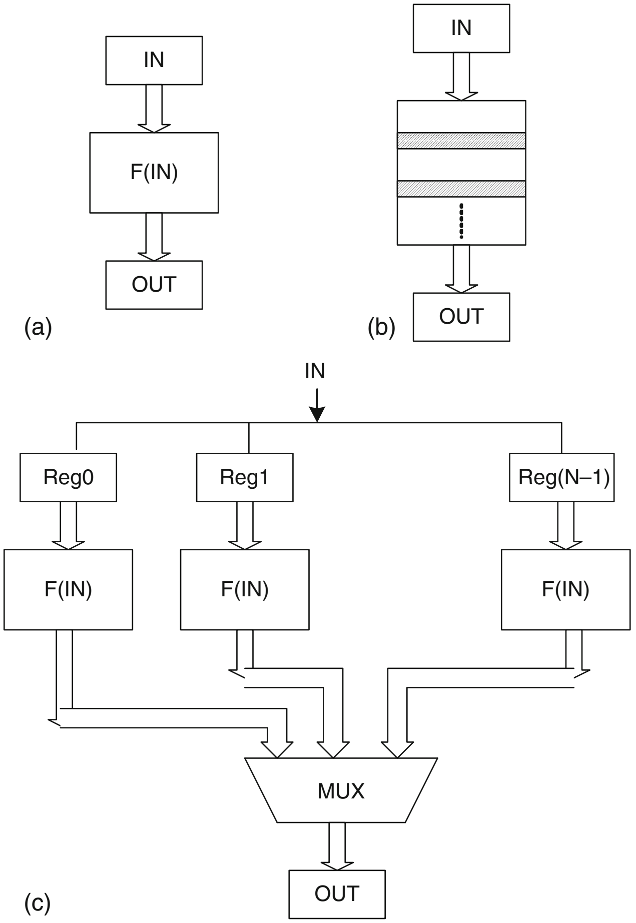


Fig. 2.15 Architectures enabling voltage scaling

idle. This saves power dissipation both in the clock tree and in the registers for when they do not need to change. Clock gating may be applied at any level of the design hierarchy, from a small logic block to the entire chip. Figure 2.16 illustrates how a register is gated. When the gating condition is true, the output of the OR gate will be 1, the D flip-flop value will not change, and therefore the following logic will not

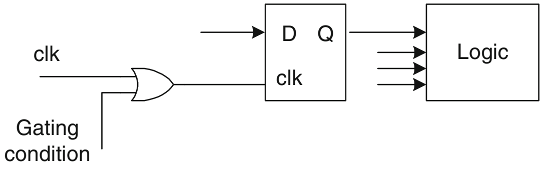


Fig. 2.16 Illustration of clock gating

switch. However, note that the more fine-grained the design is clock gated, the more complicated the control logic and hence, the more area overhead there will be.

2.4.3 Logic Level Optimization

Once various system level, architectural and technological choices are made, logic synthesis is performed to generate the netlist of gate specifications. During logic synthesis, different techniques are applied to transform the original RTL description to optimize for a specified objective function such as area, delay, power or testability. After logic synthesis, the power consumption of a circuit will be determined by the switched capacitance of the logic. In this section, a number of techniques for power estimation and minimization during logic synthesis for combinational and sequential circuits will be introduced. The strategy for low power logic synthesis is to obtain low switching activity factors and reduce the capacitive load at the nodes.

2.4.3.1 Multi-Level Network Optimization

Multi-level optimization of a Boolean network involves creating new intermediate signals and/or removing some of the existing ones to reduce area or improve performance. By adjusting the cost function targeting power consumption, multi-level network optimization techniques for low power design are also proposed [75, 76].

Don't care optimization is one way to reduce the switching activities. We use an example in Fig. 2.17 to illustrate the method. The input and output relation can be analyzed as follows: when $a = 1, f = c$; when $a = 0, f = 0$. Hence, c can only be observed at f when $a = 1$. However, when $a = 1, c = 1$. It can be seen that the circuit is equivalent to $f \equiv a$. b can be regarded as a don't care signal and assigned to be 1. The key point for don't care optimization is to identify don't care signals and minimize the switching activities of the circuits using them. Note that, when the probability for a signal to be 1, prob(1), is 0.5, its probable switching activity from 0 to 1 is the largest (prob(1) × (1 − prob(1))). Considering how changes in the global function of an internal node affects the switching activity of nodes in its fanout, [76] presents a greedy network optimization procedure. The optimization procedure works from the circuit outputs to the inputs, simplifying the fanouts of nodes. Once a node n is simplified, the procedure propagates those don't care conditions which help reduce the switching activity of its connected nodes. If the signal probability

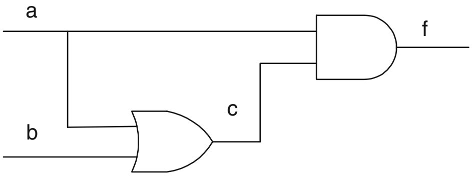


Fig. 2.17 Example for optimization with don't care conditions

of the node is greater than (less than or equal to) 0.5, don't care conditions are used to increase (decrease) the signal probability. Power consumption in a combinational logic circuit has been reduced by around 10% as a result of this optimization [10].

2.4.3.2 Logic Factorization

Sharing of common sub-expressions across the design reduces the number of literals and area, and may also improve the circuit performance. Based on the input of a set of Boolean functions, a procedure called kernel finds the common divisors to two or more functions. The best few common divisors are factored out, and the affected functions are simplified. This process is repeated until no common divisors can be found. The global area optimization process of SIS [77] has been proven to be very well suited for extensions to power optimization. The kernel extraction procedure is modified in [78] to generate multi-level circuits with minimized power.

An example of extraction is shown in Fig. 2.18. It depicts two ways of decomposition of the same function $f = ab + ac + bc$. Power consumption of decomposition A is equal to $P_{\text{decomp}(A)} = 2P_a + 2P_b + P_c + P_g + P_f$, and power consumption of decomposition B is equal to $P_{\text{decomp}(B)} = P_a + 2P_b + 2P_c + P_h + P_f$. Then $P_{\text{decomp}(A)} - P_{\text{decomp}(B)} = P_a + P_g - P_b - P_h$. According to the signal probabilities of the inputs, the decomposition with lower power consumption will be selected. However, it needs to noted that the computation for finding the optimal divisor for power minimization is expensive. Therefore, it is desirable to calculate only a subset of the candidate divisors.

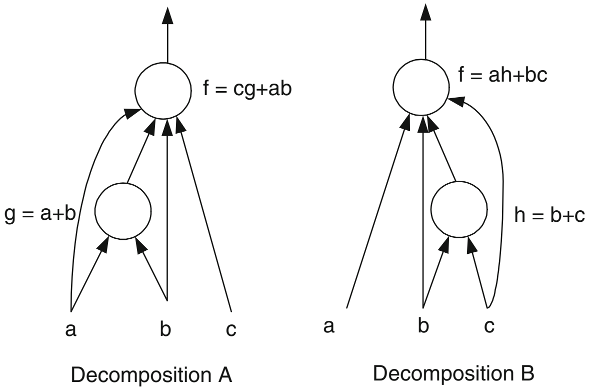


Fig. 2.18 Logic decomposition for low power

2.4.3.3 Path Balancing to Reduce Glitches

Balancing path delays reduces glitches in a circuit, which in turn reduces circuit dynamic power dissipation. This can be achieved through balanced logic structuring or delay insertion. Figure 2.19 shows an example of how logic structuring can balance an input path to reduce glitches. In (a), assuming that all the primary inputs

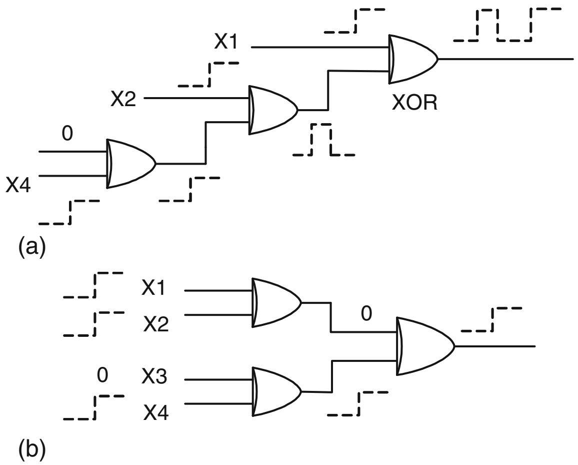


Fig. 2.19 Logic restructuring of a circuit to reduce glitches

arrive at the same time, different arrival times arise for the inputs to the XOR gates, due to the long path. This results in glitches in the internal lines and circuit output. By restructuring the logic as in (b), the input paths to the gates are balanced and such glitches are removed.

Another way to balance paths is to insert buffers to balance the path delay. Figure 2.20 illustrates one example for buffer insertion. The key issue in buffer insertion is to use the minimum number of buffers to achieve the maximum reduction in glitches.

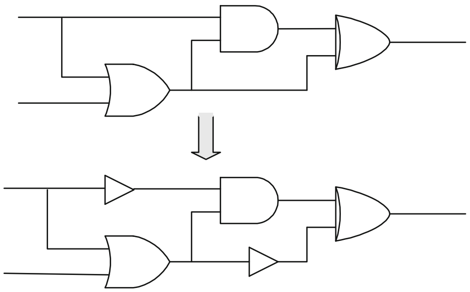


Fig. 2.20 Balance the path by inserting buffers

2.4.3.4 Technology Mapping

Technology mapping is the procedure of binding a set of logic equations (or Boolean network) to gates in the target cell library such that a given objective is optimized [77]. The objective can target area, performance, or power optimization. It can be solved by using the tree covering problem, as discussed in [79]. The circuit is decomposed into trees, and the tree covering problem is applied on all the

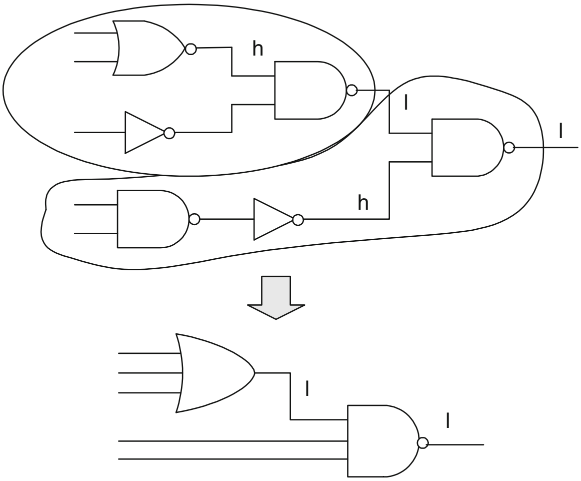


Fig. 2.21 Technology mapping for low power

trees separately. The tree covering problem can be solved efficiently with dynamic programming. The problem of minimizing average power consumption during technology mapping is addressed in [80–83]. The general principle is to hide nodes with high switching activity inside the gates that drive smaller load capacitances, as shown in Fig. 2.21. Lin and de Man [82] have considered power dissipation in the cost function for technology mapping. The formula for power estimation is given by

$$\text{Power} = \sum_{i=1}^{i=n} \alpha_i C_i f V_{\text{dd}}^2 \tag{2.25}$$

where f is the clock frequency and C_i, α_i are the load capacitance, and 0–1 switching activity is associated with the node i. The key point of dynamic programming approach is to break the optimization problem down into similar substeps that estimate the partial cost of intermediate solutions. Hence, the cost function for power dissipation during tree mapping can be formulated as

$$\text{power}(g,n) = \text{power}(g) + \sum_{n_i \in \text{inputs}(g)} \text{MinPower}(n_i) \tag{2.26}$$

where power(g) is the power contribution of choosing gate g to implement node n. The second term is the sum of minimum power costs for the corresponding subtrees rooted at the input pins of g. To minimize power consumption under a delay constraint, [80] proposes an approach composed of two steps. In the first step, the curve of power consumption versus arrival time at all nodes is computed and associated with the nodes. In the second step, a reverse pass from the root of the tree is performed to select the mapping with minimum cost that satisfies the delay constraint.

Note that under a real delay model, the dynamic programming-based tree mapping algorithm does not guarantee to find an optimal solution. The extension to a real delay is considered in [84].

2.4.3.5 State Assignment

We have discussed several methods for power optimization of combinational logic. Next, we will target power minimization in sequential logic. A finite state machine (FSM) is shown in Fig. 2.22. The first method addresses how to assign the states of an FSM to minimize the total number of state transitions (from current state S_1 to next state S_2), given the signal probability of primary inputs. The state assignment algorithm tries to minimize the activity for number of transitions at the present state inputs to the state machine. The basic assumption is that the higher the state transition activities at the input to a combinational circuit, the higher the rate at which the internal nodes of the circuit will switch, and hence more power is consumed. The approach to minimize the state transitions is to give uni-distance codes to states with high transition frequencies to one another [78]. The average number of switches at the present state inputs to a state machine can be modeled as:

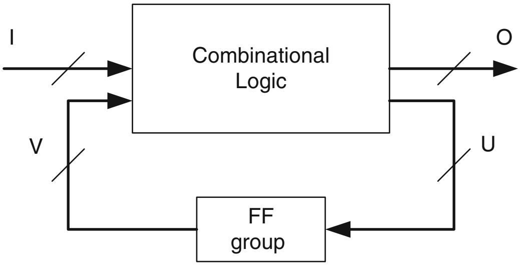


Fig. 2.22 State machine representation

$$N_{\text{swi}} = \sum_{\text{overalledges}} \text{prob}_{ij} H(S_i, S_j) \tag{2.27}$$

where prob_{ij} is the probability of transaction from state i to state j. $H(S_i, S_j)$ is the Hamming distance between state i and j. For example, the probability of state transaction for the state assignment in Fig. 2.23a is $(0.5+0.8)\times 2+0.4+0.6+0.5+0.2 = 4.3$. By changing the state assignment and reducing the Hamming distance between the states with high switching probability, the total state transactions are reduced to $(0.4+0.5)\times 2+0.5+0.8+0.6+0.2 = 3.9$ as shown in Fig. 2.23b. The optimization can be achieved by iterative solution, such as simulated annealing.

However, the above objective function ignores the power consumption in the combinational logic that implements the next state. A more effective approach [85] considers the complexity of the combinational logic resulting from the state assignment and modifies the objective functions to achieve lower power dissipation. Genetic algorithm-based approaches [86–88] are also proposed for FSM synthesis

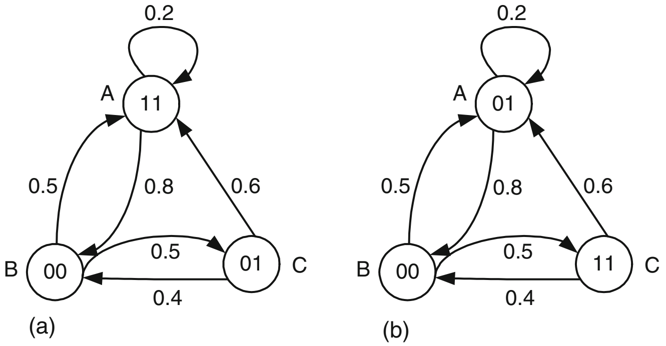


Fig. 2.23 Two different state assignments of the FSM

and state assignment, targeting low power dissipation. The approach considers both register switching and gate switching, and significant (over 10%) power savings can be obtained as compared to conventional encoding schemes, such as NOVA [89] and JEDI [90].

2.4.3.6 Precomputation

The basic idea for precomputation is to selectively compute some logic outputs and use them to reduce the internal switching activity in the succeeding clock cycle. A precomputation architecture is shown in Fig 2.24. The inputs to combinational logic have been partitioned into two sets, stored in registers R_1 and R_2. The output of the logic feeds register R_3. Boolean functions g_1, g_2 are the predictor functions. The precomputation conditions are that

$$\begin{aligned} g_1 = 1 \Rightarrow f = 1 \\ g_2 = 1 \Rightarrow f = 1 \end{aligned} \tag{2.28}$$

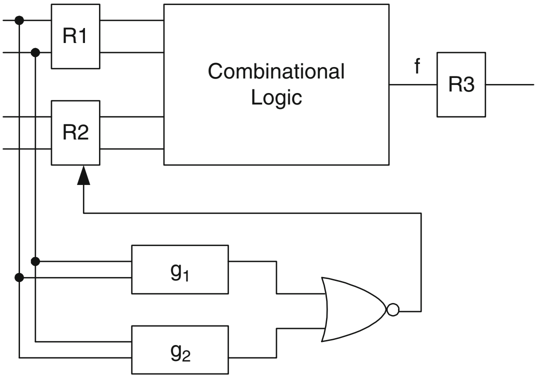


Fig. 2.24 A precomputation architecture

It means that if either g_1 or g_2 is evaluated to be 1, we can set the load enable signal of register R_2 to 0. This implies that the set of inputs will not be loaded to R_2. Since R_1 is still updated, function *f* will get the correct value. Since register R_2 and part of the combinational logic block are not switched, power reduction is obtained.

One example of usage of precomputation is the *n*-bit comparator that compares two *n*-bit numbers *A* and *B*. When the MSB $A_n > B_n \Rightarrow A > B$ and $A_n < B_n \Rightarrow A < B$. The result can be correctly predicted regardless of the left input bits.

Another way to obtain the predictor functions is to apply Shannon's decomposition. According to Shannon's decomposition theorem,

$$f = x_i f_{x_i} + x_i' f_{x_i'} \tag{2.29}$$

Define $U_{x_i} f = x_i f_{x_i} \cdot x_i' f_{x_i'}$ to be the universal quantification of *f* and x_i, and we can get the following equivalence.

$$U_{x_i} f = 1 \Rightarrow f_{x_i} = f_{x_i'} = 1 \Rightarrow f = x_i \cdot 1 + x_i' \cdot 1 = 1 \tag{2.30}$$

Similarly, it can be derived as

$$U_{x_i} f' \Rightarrow f'_{x_i} = f'_{x_i'} = 1 \Rightarrow f_{x_i} = f_{x_i'} = 0 \Rightarrow f = 0 \tag{2.31}$$

Hence, for function *f* with *m* input variables, there can be *m* predictor functions, i.e., $g_i = U_{x_i} f$. Power reductions of up to 40% have been reported in some examples. However, if the computation of a predictor is too complex, long delays may be invoked.

2.4.3.7 Retiming

Retiming is the process of re-positioning the flip-flops in a pipelined circuit in order to either minimize the number of flip-flops or minimize the delay of the longest pipeline stage. It is noted that an unbalanced path will cause glitches in combinational logic; however, the output of a flip-flop only makes one transition when the clock is asserted, as illustrated in Fig. 2.25. Based on this observation, retiming techniques are proposed that target low power dissipation [91, 92]. We can see in Fig. 2.25 that glitches are limited at the local capacitance C_K. However, since C_L is much larger than C_K, power dissipation due to glitches is greatly saved. The idea is to identify circuit nodes with high glitches and high load capacitance as proper candidates for adding flip-flops. The flip-flops also can be shifted around to help reduce the glitches at nodes with large capacitance. Figure 2.26 shows two examples of flip-flop shifting.

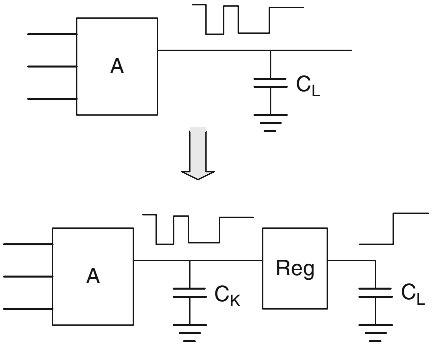


Fig. 2.25 Flip-flop insertion to minimize glitches

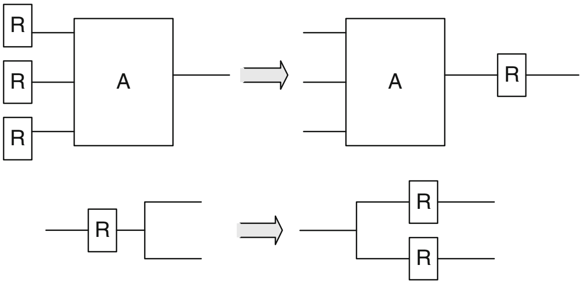


Fig. 2.26 Flip-flop shifting to reduce glitches

2.4.4 Circuit Level Optimization

In this level, circuit optimization and transistor sizing can be used to minimize power dissipation. We will first consider the circuit optimization techniques, such as equivalent pin reordering.

Equivalent pin reordering: Considering the NAND gate shown in Fig. 2.27, internal parasitic capacitance exists at the internal nodes. Suppose that the inputs of the NAND gate switch from 10 to 00. Before transition, the inputs can be

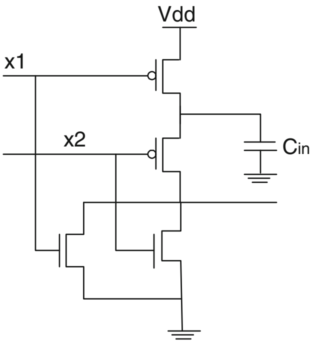


Fig. 2.27 NAND gate with internal parasitic capacitance

$x1 = 0, x2 = 1$ or $x1 = 1, x2 = 0$. Suppose $x1 = 0$ and $x2$ transfers from 1 to 0, because the first pMOS is already charged, only C_l needs charging during the transaction. However, if $x2 = 0$, and $x1$ transfers from 1 to 0, both C_l and C_{in} need to be charged. We can see that the second input assignment will consume more dynamic power during the transaction. From this example, it can be observed that if a signal switches more, it should be assigned closer to the output in order to save on power consumption.

2.4.4.1 Transistor Sizing

Another way to minimize power in circuit level is to perform transistor sizing. Transistors in the circuit can be sized to optimize for delay and power. To improve the switching speed of a particular block on the critical path, one can increase the widths of the transistors in the block. This provides increased driving current and better output transition time, and hence, smaller short-circuit power dissipation. However, it needs to be noted that increasing the transistor widths of the block also increase the loading capacitance of its preceding block, and may adversely affect the delay and short-circuit power. Thus, the issues regarding delay and power dissipation are very interlinked and so require careful analysis. On the other hand, an alternative method is to reduce the transistor size of the block in non-critical paths. In such a case, the transistor can be sized to reduce power consumption, while its increased delay is still less than the slack allowed before sizing. Hence, in this method, power reduction is achieved without affecting delay. Transistor sizing can reduce power to 5–10% of total power.

2.4.5 Power Gating

One side effect of IC dynamic power optimization is the potential increase of IC leakage. With each technology generation, circuit supply voltages must decrease accordingly to minimize IC dynamic power consumption. This unfortunately has negative impact on circuit performance. More specifically, circuit frequency, f, can be expressed in terms of supply voltage, V_{dd}, and threshold voltage V_{t}, as follows:

$$f = \frac{k(V_{\mathrm{dd}} - V_{\mathrm{t}})^{\alpha}}{V_{\mathrm{dd}}} \tag{2.32}$$

where $1 \leq \alpha \leq 2$.

The above equation shows that f decreases as V_{dd} decreases. To address circuit performance concerns, it is then important to reduce the circuit threshold voltage V_{t} accordingly, which, however, results in exponential increase in IC leakage power dissipation.

Leakage power optimization has been intensively studied, and a variety of techniques have been proposed. Among which, power gating has been one of the most effective techniques, and thus is widely adopted in low-power IC design [93–95]. Power gating minimizes IC leakage power consumption by shutting off the power

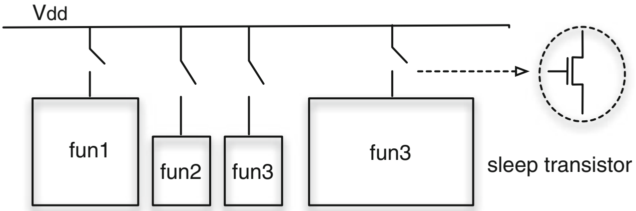


Fig. 2.28 Power gating for IC leakage power minimization

supply when a circuit is idle. As shown in Fig. 2.28, power gating implementations typically involve multi-threshold circuit designs. Circuit logic functions are implemented using high-performance, hence low V_t, technology, but are connected to the power supply through high V_t footer and/or header sleep transistors. During run-time circuit operation, idle functional units are identified, and the corresponding power supply is then shut off by turning off the sleep transistor. Since the the sleep transistor uses high V_t technology, the overall leakage current of the idle functional blocks can be substantially reduced.

However, power gating also introduces a number of design challenges. First, the design complexity and area overhead of power gating must be carefully considered. The area overhead is mainly introduced by sleep transistors and their corresponding control logic. In particular, due to stringent circuit performance requirements, sleep transistors have large footprints. In addition, to prevent floating output when a functional unit is in sleep mode, extra logic, such as weak pull-up/down devices, are required. Furthermore, ground bounce may be introduced during circuit power state transition due to parasitic capacitance charge and discharge, which has serious impact on circuit run-time performance and reliability. Addressing this problem requires careful design and placement of sleep transistors, as well as intelligent run-time wakeup scheduling. Overall, power gating is beneficial, but the design complexity and cost must be carefully considered.

2.5 Conclusions

This chapter overviewed the IC power consumption issue. It first explained the primary circuit power dissipation mechanisms, i.e., dynamic and leakage power consumption. It then described how to model and estimate IC power consumption. Finally, a wide range of recently proposed power optimization techniques were discussed. The following sections will provide more detailed discussion of various related IC power consumption issues, as well as recently proposed power management and optimization techniques. Even though low-power IC design has been an active research field over two decades, IC power consumption challenges will continue to grow, and power optimization will continue to be the one of the formost IC design objectives.

References

1. International Technology Roadmap for Semiconductors, http://public.itrs.net/, United States Semiconductor Industry Association, USA, 2007
2. Yuan CP, Trick TN (1982) A simple formula for the estimation of the capacitance of twodimensional interconnects in VLSI circuits. IEEE Electron Device Let EDL-3(12):391–393
3. Lee M (1996) A fringing and coupling interconnect line capacitance model for VLSI onchip wiring delay and crosstalk. In: IEEE international symposium on circuits and systems, Atlanta, GA, USA, pp 233–236
4. Shoji M (1988) CMOS digital circuit technology. Prentice Hall, Englewood Cliffs, NJ
5. Vanoostende P et al (1992) Evaluation of the limitations of the simple CMOS power estimation formula: comparison with accurate estimation. In: Proceedings of European workshop on power and timing modelling, Paris, France, pp 16–25
6. Bakoglu H (1987) Circuits, interconnections, and packaging for VLSI. Addison-Wesley, New York, NY
7. Pullela S, Menezes N, Omar J, Pillage L (1993) Skew and delay optimization for reliable buffered clock trees. In: Digest of technical papers of IEEE international conference on computer aided design, Santa Clara, CA, USA, pp 556–562
8. Zhu D, Dai W, Xi J (Nov 1993) Optimal sizing of high speed clock networks based on distributed RC and transmission line models. In: Digest of technical papers of IEEE international conference on computer aided design, Santa Clara, CA, USA, pp 628–633
9. Veendrick HJM (1984) Short-circuit dissipation of static CMOS circuitry and its impact on the design of buffer circuits. IEEE JSSC 19(4):468–473
10. Rabaey JM, Pedram M (1996) Low power design methodologies. Kluwer, Boston, MA
11. Bisdounis L, Koufopavlou O (1999) Analytical modeling of short-circuit energy dissipation in submicron CMOS structures. In: Proceedings of IEEE international conference on electronics, circuits and systems, Pafos, Cyprus, pp 1667–1670
12. da Costa EAC, Cardoso R, Bampi S (2000) Modeling of short circuit power consumption using timing-only logic cell macromodels. In: Proceedings 13th symposium on integrated circuits and systems design, Manaus, Brazil, pp 222–227
13. Naffziger S et al (Jan. 2006) The implementation of a 2-core, multi-threaded Itanium family processor. IEEE Solid J-State Circuits 41(1):197–209
14. Mistry K, Allen C, Auth C, Beattie B, Bergstrom D, Bost M, Brazier M, Buehler M, Cappellani A, Chau R, Choi C-H, Ding G, Fischer K, Ghani T, Grover R, Han W, Hanken D, Hattendorf M, He J, Hicks J, Heussner R, Ingerly D, Jain P, James R, Jong L, Joshi S, Kenyon C, Kuhn K, Lee K, Liu H, Maiz J, McIntyre B, Moon P, Neirynck J, Pae S, Parker C, Parsons D, Prasad C, Pipes L, Prince M, Ranade P, Reynolds T, Sandford J, Shifren L, Sebastian J, Simon D, Sivakumar S, Smith P, Thomas C, Troeger T, Vandervoorn P, Williams S, Zawadzki K (Dec 2007) A 45 nm logic technology with High-k+Metal gate transistors, strained silicon, 9 Cu interconnect layers, 193 nm dry patterning, and 100% Pb-free packaging. In: IEEE international electron devices meeting, Washington, DC, USA
15. Roy K, Mukhopadhyay S, Mahmoodi-Meimand H (Feb 2003) Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits. Proc IEEE 91(2):305–327
16. Chandrakasan A, Bowhill W, Fox F (2001) Design of high-performance microprocessor circuits. IEEE, Piscataway, NJ
17. Cao KM et al (Dec 2000) BSIM4 gate leakage model including source-drain partition. In: International Electron Device Meeting (IEDM) technology digest, San Francisco, CA, USA, pp 815–818
18. Nagata M (1992) Limitations, innovations and challenges of circuits and devices into a half micrometer and beyong. IEEE J Solid-State Circuits 27:465–472
19. Hu C (Jun 1994) MOSFET scaling in the next decade and deyong. In: Proceedings of international semiconductor device research symposium, Charlottesville, VA, USA, pp 105–114

20. Meindl JD (1995) Low power microelectronics: retrospect and prospect. Proc IEEE 83(4):619–635
21. Iwai H (1998) CMOS scaling towards its limits. In Proceedings of Solid-State and Integrated Circuit Technology, Beijing, China, pp 31–34
22. Frank DJ (Mar/May 2002) Power-constrained CMOS scaling limits. IBM J Res Dev. 46(2/3):235–244
23. Lundstrom M (Dec 2003) Device physics at the scaling limit: what matters? MOSFETs. In: Proc. IEEE Int. Electron Devices Meeting, Washington, DC, USA, pp 33.1.1–33.1.4
24. Nowak EJ, Aller I, Ludwig T, Kim K, Joshi RV, Chuang CT, Bernstein K, Puri R (Jan-Feb 2004) Turning silicon on its edge. IEEE Circuits Devices Mag 20(1):20–31
25. Mukhopadhyay S, Roy K (Aug 2003) Modeling and estimation of total leakage current in nano-scaled CMOS devices considering the effect of parameter variation. In: International symposium on low power electronics and design, Seoul, Korea, pp 172–175
26. Srivastava A, Bai R, Blaauw D, Sylvester D (2002) Modeling and analysis of leakage power considering within-die process variations. In: Int. symp. low power electronics and design, Monterey, CA, USA, pp 64–67
27. Rao R, Srivastava A, Blaauw D, Sylvester D (2003) Statistical estimation of leakage current considering inter- and intra-die process variation. In: International symposium on low power electronics and design, Seoul, Korea, pp 19–23
28. Narendra S, De V, Borkar S, Antoniadis D, Chandrakasan A (2002) Full-chip sub-threshold leakage power prediction model for sub-0.18um CMOS. In: International symposium on low power electronics and design, Monterey, CA, USA, pp 19–23
29. Agarwal K, Rao R, Sylvester D, Brown R (Jun 2007) Parametric yield analysis and optimization in leakage dominated technologies. Trans. VLSI Syst15(6):613–623
30. Rao R, Devgan A, Blaauw D, Sylvester D (2004) Parametric yield estimation considering leakage variability. Design automation conf., San Diego, CA, USA, pp 442–447
31. King SM (Oct 1986) Accurate simulation of power dissipation in VLSI circuits. IEEE J Solid State Circuits 21(5):889–891
32. Burch R, Najm FN, Yang P, Trick T (Mar 1993) A Monte Carlo approach for power estimation. IEEE Trans.VLSI syst 1(1):63–71
33. Goldstein H (Sept 1979) Controllability/observability of digital circuits. IEEE Trans. Circuits Syst 26(9):685–693
34. Lee C (Jul 1959) Representing of switching circuits by binary-decision diagrams. Bell Syst Tech J. 38:985–999
35. Bryant R (Aug 1986) Graph-based algorithms for Boolean function manipulation. IEEE Trans Comput Aided Des, C-35(8):677–691
36. Chou TL, Roy K, Prasad S (Nov 1994) Estimation of circuit activity considering signal correlations and simultaneous switching. In: Proceedings of IEEE international conference on computer-aided design, San Jose, CA, USA, pp 300–303
37. Ghosh A, Devadas S, Keutzer K, White J (Jun 1992) Estimation of average switching activity in combinational and sequential circuits. In: Proceedings of design automation conference, Anaheim, CA, USA, pp 253–259
38. Monteiro J, Devadas S, Ghosh A (Nov 1993) Retiming sequential circuits for low power. In: Proceedings of IEEE international conference on Computer Aided Design, Santa Clara, CA, USA, pp 398–402
39. Chen Z, Roy K (1998) A power macromodeling technique based on power sensitivity. In: Proceedings of design automation conference, San Francisco, CA, USA, pp 678–683
40. Gupta S, Najm FN (1997) Power macromodeling for high level power estimation. In: Proceedings of design automation conference, Anaheim, CA, USA, pp 365–370
41. Muttreja A, Raghunathan A, Ravi S, Jha NK (2004) Automated energy/performance macromodeling of embedded software. In: Proceedings of design automation conference, pp 99–102
42. Butts JA, Sohi GS (Dec 2000) A static power model for architects. In: Proceedings of international symposium on microarchitecture, Monterey, CA, USA, pp 191–201

43. Martin SM et al (Nov 2002) Combined dynamic voltage scaling and adaptive body biasing for lower power microprocessors under dynamic workloads. In: Proceedings of international conference on computer-aided design, San Jose, CA, USA, pp 721–725
44. Narendra S et al (Feb 2004) Full-chip subthreshold leakage power prediction and reduction techniques for sub-0.18 CMOS. IEEE J Solid-State Circuits 39(2):501–510
45. Tsai YF et al (Nov 2004) Characterization and modeling of run-time techniques for leakage power reduction. IEEE Trans VLSI Syst 12(11):1221–1232
46. Abdollahi A, Fallah F, Pedram M (Feb. 2004) Leakage current reduction in CMOS VLSI circuits by input vector control. IEEE Trans VLSI Syst12(2):140–154
47. BSIM4. http://www-device.eecs.berkeley.edu/bsim3/bsim4.html, Berkeley University, CA, USA, 2009
48. HSPICE. http://www.synopsys.com/products/mixedsignal/hspice/hspice.html, Synopsys, CA, USA, 2005
49. Sirichotiyakul S et al (Apr 2002) Duet: an accurate leakage estimation and optimization tool for Dual- Vt circuits. IEEE Trans VLSI Syst10(2):79–90
50. Lee D et al (Jun 2003) Analysis and minimization techniques for total leakage considering gate oxide leakage. In Proceedings of design automation conference, Anaheim, CA, USA, pp 175–180
51. Rao RM et al (Aug 2003) Efficient techniques for gate leakage estimation. In: Proceedings of international symposium on low power electronics & design, Seoul, Korea, pp 100–103
52. Rastogi A et al (Jan 2007) An efficient technique for leakage current estimation in sub 65 nm scaled cmos circuits based on loading effect. In: Proceedings of international conference on, Bangalore, India, VLSI design
53. Do MQ et al (Mar 2007) Leakage-conscious architecture-level power estimation for partitioned and power-gated sram arrays. In: Proceedings of international symposium on quality of electronic design, San Jose, CA, USA, pp 185–191
54. Gopalakrishnan C, Katkoori S (Feb 2003) An architectural leakage power simulator for vhdl structural datapaths. In: Proceedings of international symposium on VLSI circuits, Tampa, FL, USA, pp 211–212
55. Kumar A, Anis M (Mar 2006) An analytical state dependent leakage power model for FPGAs. In: Proceedings of design automation and test in Europe conference, Munich, Germany, pp 612–617
56. Naveh A et al (May 2006) Power and thermal management in the Intel CoreDuo processor. Intel Technol J 10(2), DOI: 10.1535/itj.1002.03
57. Sato T et al (Jan 2005) On-chip thermal gradient analysis and temperature flattening for SoC design. In: Proceedings of the Asia and South Pacific design and automation conference, San Diego, CA, USA, pp 1074–1077
58. Lin S-C, Banerjee K (Nov 2006) An electrothermally-aware full-chip substrate temperature gradient evaluation methodology for leakage dominant technologies with implications for power estimation and hot-spot management. In: Proceedings of international conference on computer-aided design, San Jose, CA, USA, pp 568–574
59. Zhang Y et al (May 2003) HotLeakage: a temperature-aware model of subthreshold and gate leakage for architects. University of Virginia, Technical Report, CS-2003-05
60. Su H, Liu F, Devgan A, Acar E, Nassif S (Aug 2003) Full chip leakage estimation considering power supply and temperature variations. In: Proceedings of international symposium on low power electronics & design, Seoul, Korea, pp 78–83
61. Liao WP, He L, Lepak KM (July 2005) Temperature and supply voltage aware performance and power modeling at microarchitecture level. IEEE Trans Comput Aided Des Integr Circ Syst 24(7):1042–1053
62. Galaxy design platform. http://www.synopsys.com/products/solutions/galaxyplatform.html, Synopsys, CA, USA, 2008
63. Li P, Pileggi LT, Ashghi M, Chandra R (Nov 2004) Efficient full-chip thermal modeling and analysis. In: Proceedings of international conference on computer-aided design, San Jose, CA, USA, pp 319–326

64. Zhan Y, Sapatnekar SS (Oct 2005) A high efficiency full-chip thermal simulation algorithm. In: Proc. int. conf. computer-aided design, San Jose, CA, USA
65. Huang W, Ghosh S, Velusamy S, Sankaranarayanan K, Skadron K, Stan M (May 2006) HotSpot: A compact thermal modeling methodology for early-stage VLSI design. IEEE Trans VLSI Syst14(5):501–524
66. Smy T, Walkey D, Dew S (Jul 2001) Transient 3D heat flow analysis for integrated circuit devices using the transmission line matrix method on a quad tree mesh. Solid-State Electron 45(7):1137–1148
67. Liu P, Qi Z, Li H, Jin L, Wu W, Tan S, Yang J (Oct 2005) Fast thermal simulation for architecture level dynamic thermal management. In: Proceedings of international conference on computer-aided design, San Jose, CA, USA
68. Wang T, Chen C (Dec 2002) 3-D thermal-ADI: a linear-time chip level transient thermal simulator. IEEE Trans Comput Aided Des Integr Circ Syst 21(12):1434–1445
69. Yang Y, Zhu C, Gu ZP, Shang L, Dick RP (Nov 2006) Adaptive multi-domain thermal modeling and analysis for integrated circuit synthesis and design. In: Proceedings of international conference on computer-aided design, San Jose, CA, USA
70. Murthy JY, Narumanchi SVJ, Pascual-Gutierrez JA, Wang T, Ni C, Mathur SR (2005) Review of multi-scale simulation in sub-micron heat transfer. Int J Multiscale Comput Eng 3:5–32
71. Allec N, Hassan Z, Shang L, Dick RP, Yang R (Nov 2008) ThermalScope: multi-scale thermal analysis for nanometer-scale integrated circuits. In: Proceedings of international conference on computer-aided design, San Jose, CA, USA, pp 603–610
72. Raghunathan A, Jha NK (May 1995) An ILP formulation for low power based on minimizing switched capacitance during datapath allocation. In: Proceedings of international symposium on circuits & system, Seattle, WA, USA, pp 1069–1073
73. MeLna R, Rabaey J (Apr 1994) Behavioral level power estimation and exploration. In: Proc. Int. wkshp. low power design, Napa Valley, CA, USA, pp 197–202
74. Chaudhuri S, Blythe SA, Walker RA (Sept 1995) An exact solution methodology for scheduling in a 3D design space. In: Proceedings of international symposium on system level synthesis, Cannes, France, pp 78–83
75. Wang Q, Vrudhula SBK (1996) Multi-level logic optimization for low power using local logic transformations. In: Proceedings of IEEE international conference on computer-aided design, San Jose, CA, USA, pp 270–277
76. Irnan S, Pedram M (1994) Multi-level network optimization for low power. In: Proceedings of IEEE international conference on computer-aided design, San Jose, CA, USA, pp 372–377
77. SIS. http://embedded.eecs.berkeley.edu/pubs/downloads/sis/index.htm, Berkeley University, CA, USA, 2003
78. Roy K, Prasad SC (Dec 1993) Circuit activity based logic synthesis for low power reliable operations. IEEE Trans VLSI syst1(4):503–513
79. Keutzer K (Jun 1987) DAGON: technology mapping and local optimization. In: Proceedings of design automation conference, Miami Beach, FL, USA, pp 341–347
80. Tsui C-Y, Pedram M, Despain AM (1993) Technology decomposition and mapping targeting low power dissipation. In: Proceedings of design automation conference, Dallas, Texas, USA, pp 68–73
81. Tiwari V, Ashar P, Malik S (Jun 1993) Technology mapping for low power. In: Proceedings of design automation conference, Dallas, Texas, USA, pp 74–79
82. Lin B, De man H (Oct 1993) Low-Power driven technology mapping under timing constraints. In: Proceedings of international conference on computer design, Cambridge, MA, USA, pp 421–427
83. Wang C-C, Kwan C-P (1997) Low power technology mapping by hiding hightransition paths in invisible edges for LUT-based FPGAs. In: Proceedings of international symposium on circuits and systems, Hong Kong, pp 1536–1539

84. C-Y. Tsui, Pedram M, Despain AM (Nov 1993) Efficient estimation of dynamic power dissipation under a real delay model. In: Proceedings of IEEE international conference on computer-aided design, Santa Clara, CA, pp 224–228
85. C-Y. Tsui, Pedram M, C-H. Chen, Despain AM (Nov 1994) Low power state assignment targeting two- and multi-level logic implementation. In: Proceedings of IEEE international conference on computer-aided design, San Jose, CA, USA, pp 82–87
86. Olson E, Kang SM (1994) State assignment for low-power FSM synthesis using genetic local search. In: Proceedings of IEEE custom integrated circuits conference, San Diego, CA, pp 140–143
87. Venkataraman G, Reddy SM, Pomeranz I (2003) GALLOP: genetic algorithm based low power FSM synthesis by simultaneous partitioning and state assignment. In: Proceedings of international conference on VLSI design, New Delhi, India, pp 533–538
88. Chattopadhyay S, Reddy PN (2004) Finite state machine state assignment targeting low power consumption. Proc IEE Comput Dig Tech 151(1):61–70
89. Villa T, Sangiovanni-Vincentelli A (Sept 1990) NOVA: state assignment of finite state machines for optimal two-level logic implementations. IEEE Trans Comput Aided Des Integr Circ Syst 9:905–924
90. Lin B, Newton AR (Aug1989) Synthesis of multiple-level logic from symbolic high-level description languages. In: Proceedings of IFIP international conference on VLSI, Munich, Germany, pp 187–196
91. Narayanan U, Pan P, Liu CL (1998) Low power logic synthesis under a general delay model. In: Proceedings of international symposium on low power electronics and design, Monterey, CA, USA, pp 209–214
92. Hsu Y-L, Wang S-J (2002) Retiming-based logic synthesis for low-power. In: Proceedings of international symposium on low power electronics and design, Monterey, CA, USA, pp 275–278
93. Jiang H, Marek-Sadowska M, Nassif SR (Oct 2005) Benefits and costs of power-gating technique. In: Proceedings of international conference on computer design, San Jose, CA, USA, pp 559–566
94. Hu Z, Buyuktosunoglu A, Srinivasan V, Zyuban V, Jacobson H, Bose P (Aug 2004) Microarchitectural techniques for power gating of execution units. In: Proceedings of international symposium on low power electronics and design, Newport Beach, CA, USA, pp 32–37
95. Shi K, Howard D (Jul 2006) Challenges in sleep transistor design and implementation in lowpower designs. In: Proceedings of design automation conference, San Francisco, CA, USA, pp 113–116

Part II
Circuit-Level Design Solutions

Chapter 3
Effect of Variations and Variation Tolerance in Logic Circuits

Swaroop Ghosh

Abstract Variations in process parameters affect the operation of integrated circuits (ICs) and pose a significant threat to the continued scaling of transistor dimensions. This fluctuation in device geometries might prevent them from meeting timing and power criteria and degrade the parametric yield. Process limitations are not exhibited as physical disparities only; transistors experience temporal device degradation as well. On top of it, power management techniques like voltage scaling, dual V_{TH}, further magnify the variation-induced reliability issues. On the other hand, conventional resiliency techniques like transistor upsizing and supply voltage boosting typically increase the power consumption. Low-power dissipation and process variation tolerance therefore impose contradictory design requirements. Such issues are expected to further worsen with technology scaling. To circumvent these non-idealities in process parameters, we describe two approaches: (1) variation-tolerant circuit designs and (2) circuits that can adapt themselves to operate correctly under the presence of such inconsistencies. In this chapter, we first analyze the effect of process variations and time-dependent degradation mechanisms on logic circuits. We consider both die-to-die and within-die variation effects. Next, we provide an overview of variation-tolerant logic design approaches. Interestingly, these resiliency techniques transcend several design abstraction levels – however in this chapter, we focus on circuit level techniques to perform reliable computations in an unreliable environment.

3.1 Introduction

Successful design of digital integrated circuits has relied on optimization of various design specifications such as silicon area, speed, and power dissipation. Such a traditional design approach inherently assumes that the electrical and physical

S. Ghosh (✉)
Intel Corporation, Portland, OR, USA
e-mail: swaroopad.ghosh@intel.com

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_3,

properties of transistors are deterministic and hence predictable over the device lifetime. However, with the silicon technology entering the sub-65 nm regime, transistors no longer act deterministically. Fluctuation in device dimensions due to manufacturing processes (sub-wavelength lithography, chemical mechanical polishing, etc.) has emerged as a serious issue in nanometer technologies. In deep sub-micron (approximately up to 0.35 μm) technology nodes, process variation was inconsequential for the IC industry. Circuits were mostly immune to minute fluctuations in geometries because the variations were negligible compared to the nominal device sizes. However, with the growing disparity between feature size and optical wavelengths of lithographic processes at scaled dimensions (below 90 nm), the issue of parameter variation is becoming severe. Variations can have two main components: inter-die and intra-die. Parametric variations between dies that come from different runs, lots, and wafers are categorized into *inter-die variations* whereas variations of transistor strengths within the same die are defined as *intra-die variations*. Fluctuations in length (L), width (W), oxide thickness (T_{OX}), flat-band conditions, etc. give rise to inter-die process variations, whereas line edge roughness (LER) or random dopant fluctuations (RDF) cause intra-die random variations in process parameters [1–7]. Systems that are designed without consideration to process variations fail to meet the desired timing, power, stability, and quality specifications. For example, a chip designed to run at 2 GHz of speed may run only at 1 GHz (due to increased threshold voltage of the devices) making it unworthy to be useful. However, one can certainly follow an overly pessimistic design methodology (using large guard-bands) that can sustain the impact of variations in all process corners. Such designs are usually time and area inefficient to be beneficial. There can be two ways to address the issue of process variation – by controlling the existing process technology (which result in less variation) or by designing circuits and architectures that are immune to variations. In this chapter, the emphasis has been given to the second choice because controlling the process is expensive and in some cases may not be viable without changing the devices itself (it should be noted that some of the emerging devices may suffer from increased variations as well [8–13]). Therefore, process variation awareness has become an inseparable part of modern system design.

3.1.1 Effect of Power and Process Variation in Logic Circuits

Apart from process variations, modern circuits also suffer from escalating power consumption [14]. Not only dynamic power but leakage power has also emerged as a dominant component of overall power consumption in scaled technologies. This is also described in Chapter 2. Power management techniques, e.g., supply voltage scaling, power gating, and multiple-V_{DD}, V_{TH} designs further magnify the problems associated with process-induced variations. For example, consider the supply voltage scaling. As the voltage is scaled down, the path delay increases worsening the effect of variations. This is elucidated in Fig. 3.1. It clearly shows that lower V_{DD}

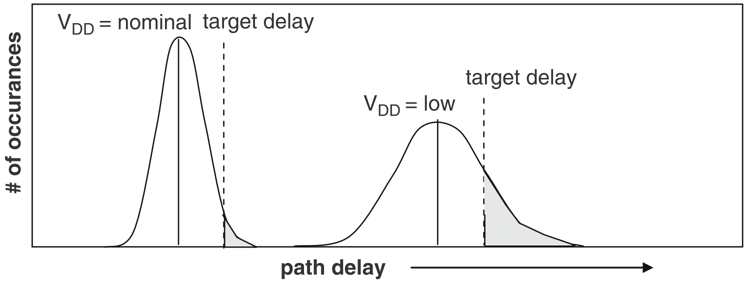


Fig. 3.1 Impact of supply voltage scaling on path delay distribution. Both mean and sigma of delay distribution as well as the number of paths failing to meet target frequency increases

not only increases the mean but also the standard deviation (STD) of the overall path delay distribution. Therefore, the number of paths failing to meet the target speed also increases thereby degrading the timing yield. On the other hand, upscaling the supply voltage reduces variation at the cost of power consumption. Power and process variation resilience are therefore conflicting design requirements and one comes at the cost of other. Meeting a desired power specification with certain degree of process tolerance is becoming an extremely challenging task.

3.1.2 Sources of Variations (Spatial vs. Temporal)

Figure 3.2 shows the taxonomy of variations in nanoscaled logic circuits. Inter-(die-to-die) and intra-die (within-die) variations fall under *spatial* (or $t = 0$) process variation. These types of variations induce speed marginalities in static logic whereas in dynamic logic they reduce the noise margin. *Temporal* variations can be categorized into two parts – environmental and aging related. Temperature and

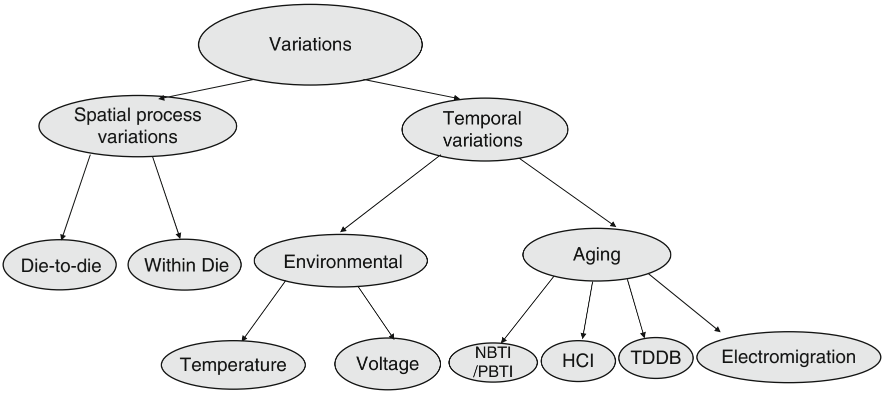


Fig. 3.2 Taxonomy of process variation

voltage fluctuations fall under the category of *environmental* variations whereas Negative Bias Temperature Instability (NBTI) [15–33], Positive Bias Temperature Instability (PBTI) [34, 35], Hot Carrier Injection (HCI) [36–38], Time-Dependent Dielectric Breakdown (TDDB) [39–41], and electromigration [42] give rise to so-called *aging* variation. Environmental variation like voltage fluctuation is the outcome of circuit switchings and lack of strong power grid. Similarly, temperature fluctuation is the result of excessive power consumption by a circuit that generates heat whereas the package lacks capability to dissipate it. Such variations degrade the robustness of the circuit by increasing speed margins inducing glitches, creating non-uniform circuit delays and thermal runaways. The aging-related variations degrade device strength when a device is used for a long period of time. For example, an IC tested and shipped for 2 GHz of speed may fail to run at the rated speed after 1 or 2 years. Along with device-related variations, the interconnect parameter variations [43] also add to the adversity of the situation.

3.1.3 Impact of Variation on Logic

The variation in L, W, T_{OX}, dopant concentration, etc. modifies the threshold voltage of the devices. One manifestation of statistical variation in V_{TH} is variation of speed between different chips. Hence, if the circuits are designed using nominal V_{TH} of transistors to run at a particular speed, some of them will fail to meet the desired frequency. Such variation in speed would lead to parametric yield loss. In dynamic logic, the parametric variations reduce the noise margin. Another detrimental effect of process variation is leakage. Statistical variation in transistor parameters results in significant spread in different components of leakage. A side effect of increased dynamic and leakage power is localized heating of the die – known as hot-spot. The hot-spots are one of the primary factors behind reliability degradation and thermal runaways.

3.1.4 Overview of Variation Insensitive Designs

In order to address the above-mentioned issues in logic circuits, two approaches can be followed: (1) design time techniques and (2) post-silicon adaptation. The design time techniques include statistical design that has been investigated as an effective method to ensure certain yield criteria. In this approach, the design space is explored to optimize design parameters (e.g., power and reliability) in order to meet timing yield and target frequency. Several gate-level sizing and/or V_{TH} (transistor threshold voltage) assignment techniques [44–55] have been proposed recently addressing the minimization of total power while maintaining the timing yield. Timing optimization to meet certain performance/area/power usually results in circuits with many equally long (critical) paths creating “a wall” in the path delay distribution. Therefore, a number of paths get exposed to process fluctuation induced delay variation. An uncertainty-aware design technique proposed in [56] describes an optimization process to reduce the wall of equally long paths. A detailed analysis of this approach has been presented in the next chapter.

On the other end of the spectrum, design techniques have been proposed for post-silicon process compensation and process adaptation to deal with process-related timing failures. Adaptive body biasing (ABB) [57, 58] is one such technique, which tries to reduce the frequency and leakage spread, improving the timing yield. Several other research works try to optimize power and yield by mixing gate sizing, multi-V_{TH} with post-Si tuning (e.g., body bias) [59–64]. Due to the superlinear dependence of dynamic power of a circuit on its operating voltage, supply voltage scaling has been extremely effective in reducing the power dissipation. Researchers have investigated logic design approaches that are robust with respect to process variations and, at the same time, suitable for aggressive voltage scaling. One such technique, called RAZOR [65–69], uses dynamic detection and correction of circuit timing errors to tune processor supply voltage. Adaptive (or variable) latency techniques for structures such as caches [70] and combined register files and execution units [71–73] can address device variability by tuning architectural latencies. The fast paths are exercised at rated frequency whereas slower paths due to process variability are operated with extended latency. In [74], Tiwari et al. presented pipelined logic loops for the entire processor using selectable "donor stages" that provide additional timing margins for certain loops. Therefore, extra delay in computation due to process variation in one stage can be mitigated by stealing some extra time from another stage.

It can be noted that pipelined design can experience timing slack either due to frequency or due to supply voltage under process variation. Therefore, both voltage and latency can be selectively picked for each of the pipelined stages for achieving optimal power and performance under variability. In [75–76], the authors describe ReVIVaL – which is a post-fabrication voltage interpolation technique that provides an "effective voltage" to individual blocks within individual processor cores. By coupling variable-latency with voltage interpolation, ReVIVaL provides significant benefits in terms of process resilience and power over implementing variable-latency alone. These architectural techniques for process resilience have been covered in Chapter 7.

Contrary to RAZOR, Ghosh et al. [77] proposed a design time technique called CRISTA to allow voltage over-scaling while meeting the desired frequency and yield. CRISTA isolates the critical (long) paths of the design and provides an extra clock cycle for those paths. The CRISTA design methodology ensures the activation of long paths to be rare, thereby, reducing performance impact. This allows them to drop the supply voltage to expose the timing slack of off-critical (short) paths. Algorithmic noise-tolerance (ANT) [78] is another energy-efficient adaptive solution for low-power broadband communication systems. The key idea behind ANT is to permit errors to occur in a signal processing block and then correct it via a separate error control (EC) block. This allows the main DSP to operate below the specified supply voltage (over-scaled voltage). The important point to note is that proper measures at all levels of hierarchy (from circuits to architecture) is essential to mitigate the problems associated with parameter variations and to design robust systems.

This chapter presents an overview of the process- and reliability-induced variations. We review several state-of-the-art circuit level adaptive techniques to deal with these issues. The chapter is organized as follows. The impact of process variation and reliability degradations is introduced in Section 3.2. The process variation-tolerant design techniques are presented in Section 3.3 whereas resilience to temporal degradation is discussed in Section 3.4. Finally, the conclusions are drawn in Section 3.5.

3.2 Effect of Parameter Variations on Logic – Failures and Parametric Yield

As mentioned before, process variations can be categorized into two broad areas: (a) spatial and (b) temporal. This is further clarified in Fig. 3.3 with an example of chips that are designed for a particular speed. Variations in device strengths between chips belonging to different runs, lots, and wafers result in dies that vary in speed. The figure shows that the speed follows a distribution where each point belongs to chips running at a particular speed. This behavior corresponds to t=0 s. Devices also change their characteristic over time due to NBTI (Negative Bias Temperature Stability), PBTI (Positive Bias Temperature Stability), TDDB (Time-Dependent Dielectric Breakdown), HCI (Hot Carrier Injection), etc. For example, the PMOS transistor becomes slower due to NBTI-induced V_{TH} degradation and wire delay may increase due to electromigration of metal. Therefore, the dies become slow and the entire distribution shifts in temporal fashion (Fig. 3.3). The variation in device characteristics over time (i.e., at $t = t'$) is termed as temporal variations.

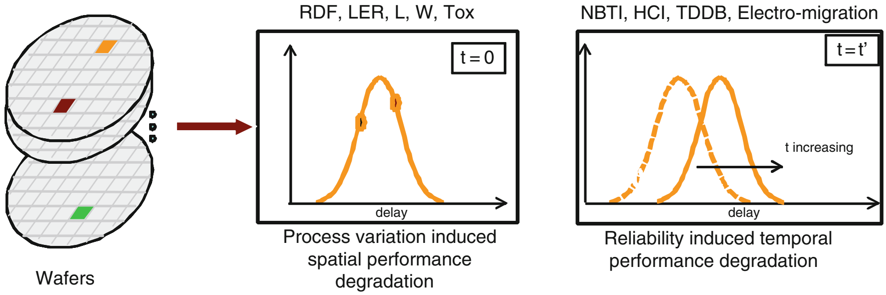

RDF, LER, L, W, Tox
NBTI, HCI, TDDB, Electro-migration
t = 0
t = t'
t increasing
delay
delay
Wafers
Process variation induced spatial performance degradation
Reliability induced temporal performance degradation

Fig. 3.3 Spatial and temporal process variation effect on circuit delay. The PDF of chip delay due to temporal degradation has been shown

3.2.1 Impact of Spatial Process Variation

The variation in L, W, T_{OX}, dopant concentration, work function, flat-band condition, etc. modifies the threshold voltage of the devices. These are termed as die-to-die V_{TH} variation. Another source of variation originates from line edge roughness, line width variation, and random variation in dopant atoms in the channel, and they are known as within-die process variation. In older technology generations, the fraction of within-die variation used to be small compared to die-to-die counterpart. However, with scaling of device dimensions within-die parametric variation has become relatively larger fraction. This is shown in Fig. 3.4. In the following paragraphs, we will discuss the manifestation of various types of spatial process variations.

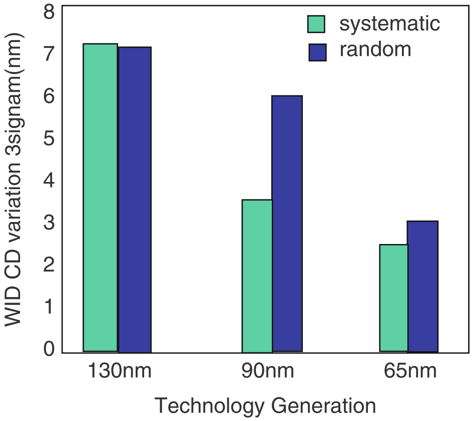


Fig. 3.4 Growing proportion of within-die process variation with technology scaling [58]

3.2.1.1 Increased Delay/Delay Distribution

One manifestation of statistical variation in V_{TH} is variation of speed between different chips. Hence, if the circuits are designed using nominal V_{TH} transistors to run at a particular speed, some of them will fail to meet the desired frequency. Such variation in speed would lead to parametric yield loss. Figure 3.5 shows how the variation in threshold voltage (under the influence of both within-die and die-to-die) translates into speed distribution. The path delay distribution can be represented by normal distribution or by a more accurate model like lognormal distribution. Several statistical static timing analysis techniques have been investigated in the past [79–83] to accurately model the mean and standard deviation (STD) of the circuit delay. The chips slower than the target delay have to be discarded (or can be sold at a lower price). An interesting observation is the impact of within-die and die-to-die process variation. It has been shown in [58] that the effect of within-die process variations tends to average out with the number of logic stages. This is shown in Fig. 3.6. However, the demand for higher frequency necessitates deeper pipeline design (i.e., shorter pipeline depths) making them susceptible to within-die process variation as well.

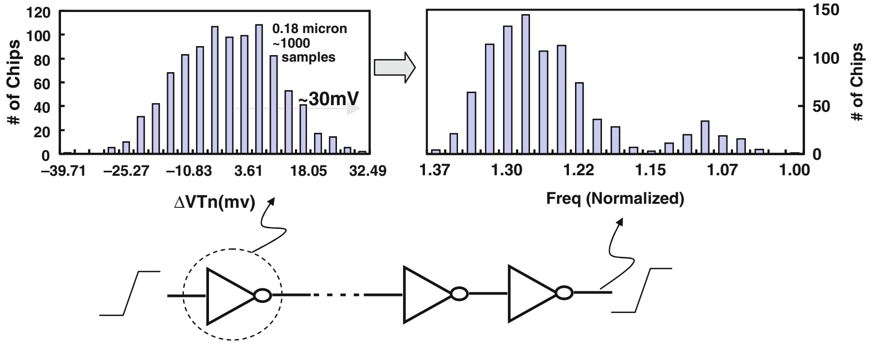


Fig. 3.5 Variation in threshold voltage and corresponding variation in frequency distribution [58]

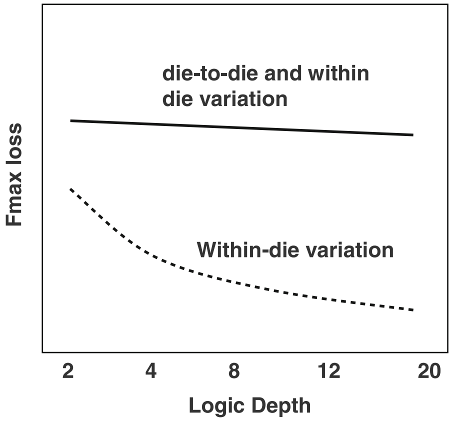


Fig. 3.6 Impact of within-die process variation becomes prominent with increasing pipeline depth [58]

3.2.1.2 Lower Noise Margins

The variations have different impacts on dynamic circuits. These circuits operate on the principle of *pre-charge* and *evaluate*. The output is pre-charged to logic “1” in the negative phase of the clock (clk = 0). The positive phase of the clock (clk = 1) allows the inputs decide if the output will be kept pre-charged or will be discharged to ground. Since the information is saved as charge at the output node capacitor, dynamic logic is highly susceptible to noise and timings of input signals. Due to inherent nature of the circuit, a slight variation in transistor threshold voltage can kill the logic functionality. For example, consider a domino logic shown in Fig. 3.7. If the V_{TH} of the NMOS transistors in the second stage is low due to process variation, then a small change in Out_1, IN_4, or IN_5 can turn the pull down path ON and may result in wrong evaluation of Out_2.

In register files, increased leakage due to lower V_{TH} dies has forced the circuit designers to upsize the keeper to obtain an acceptable robustness under worst-case

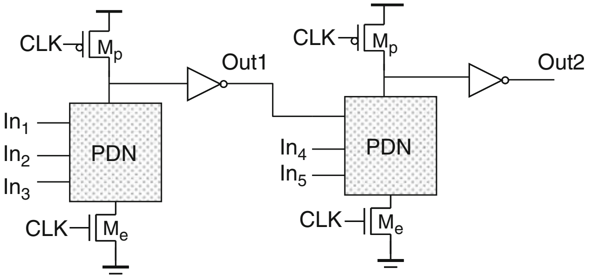


Fig. 3.7 Example of a dynamic logic circuit. The V_{TH} of NMOS transistor determines the noise margin

V_{TH} conditions. Large variation in die-to-die V_{TH} indicates that (i) a large number of low leakage dies suffer from the performance loss due to an unnecessarily strong keeper, while (ii) the excess leakage dies still cannot meet the robustness requirements with a keeper sized for the fast corner leakage.

3.2.1.3 Degraded Yield in Pipelined Design

Increasing inter-die and intra-die variations in the process parameters, such as channel length, width, threshold voltage, result in large variation in the delay of logic circuits. Consequently, estimating circuit performance and designing high-performance circuits with high yield (probability that the design will meet certain delay target) under parameter variations have emerged as serious design challenges in sub-100 nm regime. In the high-performance design, the throughput is primarily improved by pipelining the data and control paths. In a synchronous pipelined circuit, the throughput is limited by the slowest pipe segment (i.e., segment with maximum delay). Under parameter variations, as the delays of all the stages vary considerably, the slowest stage is not readily identifiable. The variation in the stage delays thus result in variation in the overall pipeline delay (which determines the clock frequency and throughput). Traditionally, the pipeline clock frequency has been enhanced by (a) increasing the number of pipeline stages, which, in essence, reduces the logic depth and hence, the delay of each stage; and (b) balancing the delay of the pipe stages, so that the maximum of stage delays are optimized. However, it has been observed that if intra-die parameter variation is considered; reducing the logic depth increases the variability (defined as the ratio of standard deviation and mean) [58]. Since the pipeline yield is governed by the yield of individual pipe stages, balancing the pipelines may not always be the best way to maximize the yield. This makes the close inspection of the effect of pipeline balancing on the overall yield under parameter variation an extremely important task.

3.2.1.4 Increased Power

Another detrimental effect of process variation is variation in leakage. Statistical variation in transistor parameters results in significant spread in different components of leakage. It has been shown in [5] that there can be ~100X variation in

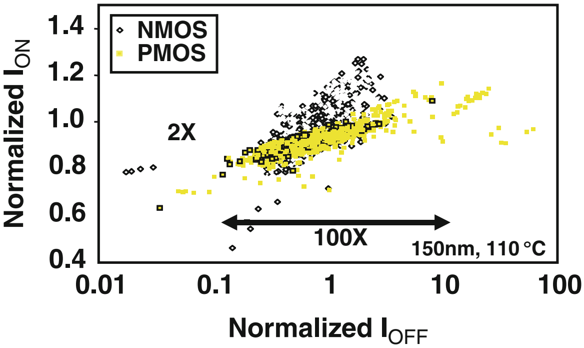


Fig. 3.8 Leakage spread in 150 nm technology (source: Intel Inc)

leakage current in 150-nm technology (Fig. 3.8). Designing for the worst case leakage causes excessive guard-banding, resulting in lower performance. On the other hand, underestimating leakage variation results in low yield, as good dies are discarded for violating the product leakage requirement. Leakage variation models have been proposed in [84–87] to estimate the mean and variance of the leakage distribution. In [87], the authors provide a complete analytical model for total chip leakage considering random and spatially correlated components of parameters, sensitivity of leakage currents with respect to transistor parameters, input vectors, and circuit topology (spatial location of gates, sizing, and temperature).

3.2.1.5 Increased Temperature

A side effect of increased dynamic and leakage power is localized heating of the die – called hot-spot. The hot-spot is outcome of excessive power consumption by the circuit. The power dissipates as heat and if the package is unable to sink the heat generated by the circuit, then it is manifested as elevated temperature. The hot-spots are one of the primary factors behind reliability degradation and thermal runaways. There have been several published efforts in compact and full-chip thermal modeling and compact thermal modeling. In [88], the authors present a detailed die-level transient thermal model based on full-chip layout, solving temperatures for a large number of nodes with an efficient numerical method. The die-level thermal models in [89] and [90] also provide the detailed temperature distribution across the silicon die. A detailed full-chip thermal model proposed in [91] uses an accurate three-dimensional (3-D) model for the silicon and one-dimensional (1-D) model for the package. A recent work [92] describes HotSpot, a generic compact thermal modeling methodology for VLSI systems. HotSpot consists of models that consider high-level interconnects, self-heating power, and thermal models to estimate the temperatures of interconnect layers at the early design stages.

3.2.2 Impact of Temporal Variations

Temporal variations like voltage and temperature fluctuations add to the circuit marginalities. The higher temperature decreases the V_{TH} (good for speed) but reduces the ON current – reducing the overall speed of the design. Similarly, if a circuit that is designed to operate at 1 V works at 950 mV due to voltage fluctuations, then the circuit speed would go below the specified rate. Since the voltage and temperature depends on the operating conditions, the circuit speed becomes unpredictable.

The aging-related temporal variations on the other hand affect the circuit speed systematically over a period of time. In random logic, the impact of NBTI degradation is being manifested as the increase of delays of critical timing paths [19, 22, 23, 26, 27, 29–33] and reduction of subthreshold leakage current [31]. In [25, 32], the authors developed a compact statistical NBTI model considering the random nature of the Si–H bond breaking in scaled transistors. They observed that from the circuit level perspective, NBTI variation closely resembles the nature of RDF-induced V_{TH} variation in a sense that it has a complete randomness even among the transistors that are closely placed in a same chip. Hence, they considered both the RDF- and NBTI-induced V_{TH} variation (σ_{RDF} and σ_{NBTI}, respectively) as follows,

$$\sigma_{V_{\mathrm{t}}} = \sqrt{\sigma_{\mathrm{RDF}}^2 + \sigma_{\mathrm{NBTI}}^2(t)} \tag{3.1}$$

where σ_{VTH} represents the total V_{TH} variation after time t.

Figure 3.9a represents the histogram of a simple inverter delay with/without the impact of NBTI variation assuming 3-year stress (for 32-nm PTM [93] using Monte Carlo and Equation (3.1)). As can be observed from the two curves on the right, the variability of gate delay can increase significantly with an added impact of NBTI. Comparison between 32 and 22 nm node results emphasizes the fact that NBTI variations will grow larger in scaled technology. It is important to note that in reality,

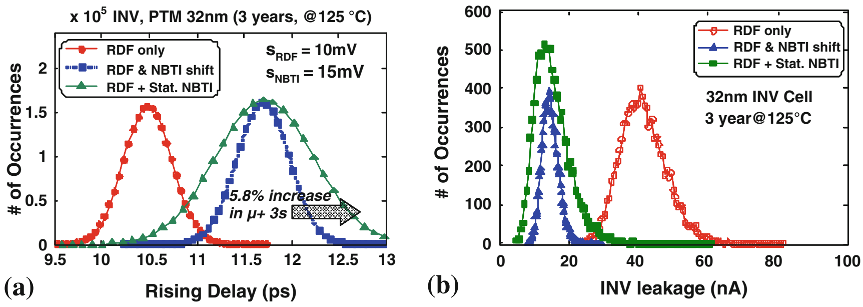


Fig. 3.9 Variation under NBTI degradation and RDF (**a**) both the mean and spread of inverter gate delay distribution increases due to combined effect of RDF and NBTI. (**b**) inverter leakage is reduced with NBTI however, the spread of leakage can increase due to statistical NBTI effect

the variation of circuit delays are mostly dominated by lower granularity sources such as inter-die variations. And as a result, random V_{TH} variation-induced by NBTI degradation will usually take only a small portion of the overall delay variations [94]. Similar effects can be observed in the subthreshold leakage variation under NBTI degradation. Figure 3.9b represents the histogram plot of inverter leakage. As can be observed, leakage current reduces due to the NBTI. However, the two curves on the left side show that the increased variability of V_{TH} due to NBTI can lead to more variation in leakage.

3.3 Variation-Tolerant Design

In the previous section, we discussed failures and yield loss due to variations. This section will provide some advanced circuit level techniques for resilience to parameter variations. These techniques can be broadly categorized as design-time (pre-Si) and run-time (post-Si) techniques. Figure 3.10 shows the taxonomy of variation-resilient schemes. The design time techniques are further categorized into (a) conservative design, (b) statistical design (Gate sizing [52–55], dual V_{TH} assignment [44–51], pipeline unbalancing [95]), and (c) resilient design (CRISTA [77]). Adaptive body biasing [57], adaptive voltage scaling, programmable sizing [96], sensor-based design [97–99], RAZOR [65–69], etc. fall under the category of post-Silicon techniques. In the following subsection, we present these methodologies in detail.

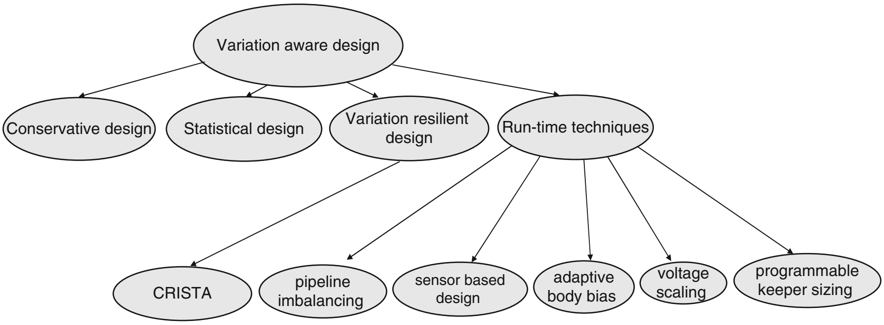


Fig. 3.10 Taxonomy of variation resilient circuit design

3.3.1 Conservative Design

Conservative approach to design circuits is based on providing enough timing margins for uncertainties, e.g., process variation, voltage/temperature fluctuations, and temporal degradation (T_{m} as shown in Fig. 3.11a). One possible option is to

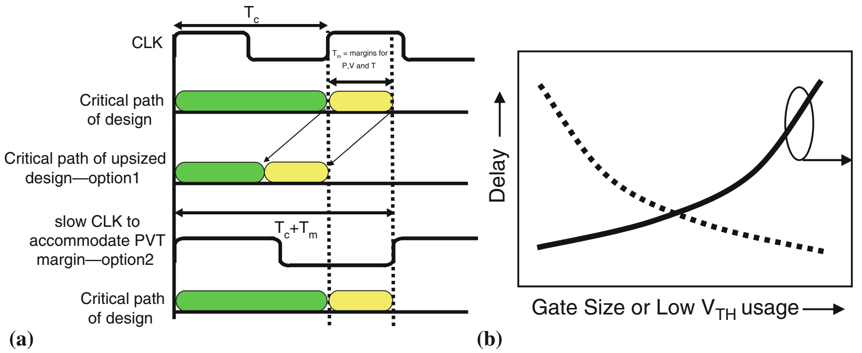

T_c
CLK
T_m = margins for P,V and T
Critical path of design
Critical path of upsized design—option1
slow CLK to accommodate PVT margin—option2
T_c+T_m
Critical path of design
(a)
Delay
Power
Gate Size or Low V_{TH} usage
(b)

Fig. 3.11 (**a**) Two possible options to tolerate process, voltage and temperature fluctuation induced timing uncertainties namely, upsizing of devices and slow down of clock frequency, (**b**) tradeoff between power and delay in conservative design approach

conservatively size the circuit to meet the target frequency after adding all margins or boosting up the supply voltage. This is shown in Fig. 3.11a as option-1. However, from Fig. 3.11b, it is obvious that conservative approach is area and power intensive. Both supply voltage and circuit size upscaling increases the power consumption. One can definitely trade power/area by slowing down the clock frequency (option-2). However, this upfront penalty cannot be accepted in today's competitive market where power and operating frequency are dominating factors. From the above discussions, it is apparent that conservative design approach is not very desirable in scaled technologies.

3.3.2 Statistical Design

3.3.2.1 Logic Sizing

It is well known that the delay of a gate can be manipulated by modifying its size [52–55]. Either the length or width of the gate can be tweaked to modulate the (W/L) ratio and control its drive ability. Since process variations may increase the delay of the circuit, many design time transistor sizing algorithms have been proposed in [52–55] to reduce the mean and STD of delay variations. The next chapter presents an in-depth analysis of various sizing techniques.

3.3.2.2 Sizing and Dual V_{TH}

Threshold voltage of the transistor is another parameter that can be adjusted to achieve a tradeoff between speed and leakage [44–51]. A new statistically aware dual-V_{TH} and sizing optimization has been suggested in [51] that considers both the variability in performance and leakage of a design. Further details on simultaneous gate sizing and dual V_{TH} can be found in the next chapter.

3.3.2.3 Pipeline Unbalancing

Shrinking pipeline depths in scaled technologies makes the path delay prone to within-die variations [81, 95]. By overdesigning the pipeline stages, better yield can be achieved at the cost of extra area and power. A framework to determine the yield of the pipeline under the impact of process variation has been proposed in [81]. This framework has been successfully employed to design high yield pipeline while compromising the area/power overhead [95] by utilizing the concept of pipeline unbalancing. This technique advocates slowing down certain pipeline stages (by downsizing the gates) and accelerating the other stages (by upsizing) to achieve overall better pipeline yield under iso-area. For example, consider a three-stage pipeline design. Let us also assume that the combinational logic of each stage is optimized for minimum area for a specific target pipeline delay and yield. If the stage yield is y, then pipeline yield $= (y)^3$. Now if the pipeline is carefully optimized to achieve the pipeline stages yields to be y_0, y_1, and y_2 (under iso-area), one can improve yield if $(y_0 y_1 y_2) > (y)^3$. This can be elucidated by assuming that the sizing of pipeline stages are done such that $y = 0.9$ and $y_0 = 0.98$, $y_1 = 0.95$, and $y_2 = 0.85$. It is obvious that better pipeline yield can be achieved by pipeline unbalancing (yield = 79.1%) rather than balanced pipeline (yield=72.9%).

3.3.3 Resilient Design

3.3.3.1 CRISTA

It is a novel paradigm for low-power, variation, and temperature tolerant circuits and system design, which allows aggressive voltage over-scaling at rated frequency. The CRISTA design principle [77] (a) isolates and predicts the paths that may become critical under process variations, (b) ensures (by design) that they are activated rarely, and (c) avoids possible delay failures in such long paths by adaptively stretching the clock period to two-cycles. This allows the circuit to operate at reduced supply voltage while achieving the required yield with small throughput penalty (due to rare two-cycle operations). The concept of CRISTA is shown in Fig. 3.12a with example of three pipelined instructions where the second instruction activates the critical path. CRISTA can be performed by gating the third clock pulse during the execution of second instruction for correct functionality of the pipeline at scaled supply. The regular clock and CRISTA-related clock-gating is shown in Fig. 3.12a for the sake of clarity. The CRISTA design methodology is applied to random logic by hierarchical Shannon expansion and gate sizing (Fig. 3.12b). Multiple expansions reduce the activation probability of the paths. In the example shown in Fig. 3.12b, critical paths are restricted within CF_{53} (by careful partitioning and gate sizing). The critical paths are activated only when $\mathrm{x}_1!\mathrm{x}_2\mathrm{x}_3 = 1$ with activation probability of 12.5% assuming that each signal can be logic "1" 50% of the time. CRISTA allows aggressive supply voltage scaling to improve power consumption by ∼40% with only 9% area and small throughput penalty for a two-stage pipelined ALU

[100] compared to standard approach to designing circuits. Note that CRISTA can be applied to circuits as well as micro-architecture level [101].

Figure 3.12c shows application of CRISTA for dynamic thermal management in a pipelined processor. Since execution (EX) unit is statistically found to be one of the hottest components, CRISTA is employed to allow voltage over-scaling in EX stage. The critical paths of EX stage are predicted by decoding the inputs using a set of pre-decoders. At nominal temperature, the pre-decoders are disabled; however, at elevated temperatures the supply voltage of the execution unit is scaled down and pre-decoders are enabled. This ensures cooling down of the system and occasional two-cycle operations in EX stage (at rated frequency) whenever the critical path is activated. If the temperature still keeps rising and crosses the emergency level then conventional dynamic voltage frequency scaling (DVFS) is employed for throttling the temperature down. DVFS is often associated with stalling while the PLL frequency is locked at the desired level. It is interesting to note that CRISTA can avoid triggering of DVFS for all SPEC2000 benchmark programs, saving throughput loss. This is evident from Fig. 3.12d that shows activation count of DVFS with and without CRISTA.

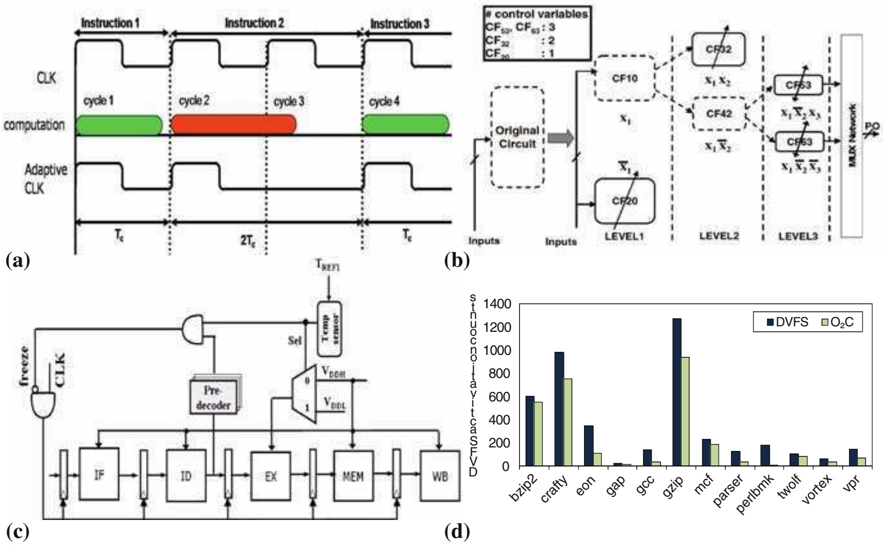


Fig. 3.12 (**a**) Timing diagram of CRISTA paradigm. Long paths are activated rarely and they are evaluated in two cycles. (**b**) Shannon expansion based CRISTA design methodology for random logic. Multiple expansions reduce the activation probability of the paths. In this example, critical paths are restricted within CF_{53} (by careful partitioning and gate sizing) which is activated when $x_1!x_2x_3 = 1$ with activation probability of 12.5%. (**c**) CRISTA at micro-architectural level of abstraction for adaptive thermal management [77]. (**d**) Activation count of dynamic voltage frequency scaling (DVFS) and CRISTA. CRISTA avoids application of DVFS saving valuable overhead in terms of throughput

3.3.4 Run Time Techniques

3.3.4.1 Adaptive Body Bias

As discussed before, threshold voltage of the transistor can be tuned carefully to speed up the design [58]. However, lowering the V_{TH} also changes the on-current. It has been noted that transistor V_{TH} is a function of body to source voltage (V_{BS}). Hence body potential of the transistor can be modulated to achieve speed or lower the leakage power. The forward body bias to PMOS transistors improves the performance due to lower V_{TH} while the reverse body bias reduces leakage due to higher V_{TH}. In [58], the authors propose adaptive body biasing technique on selective dies to improve the power and performance. The faster and leakier dies are reverse body biased while slower dies are forward biased. This results in tighter delay distribution as shown in Fig. 3.13a.

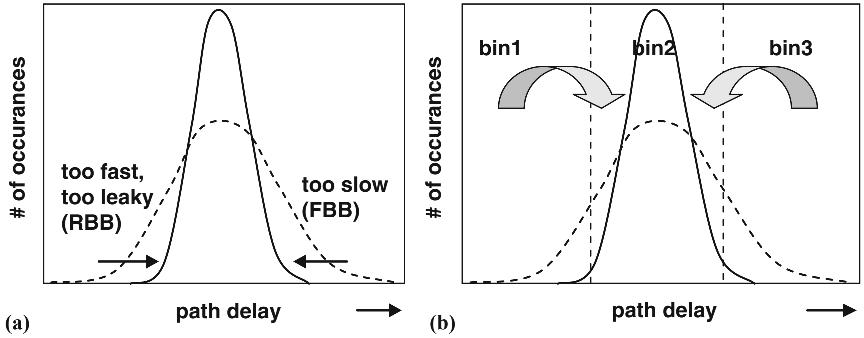


Fig. 3.13 (**a**) Effect of adaptive body bias – slower paths are forward body biased whereas faster paths are reversed body biased to squeeze the distribution. (**b**) Effect of adaptive voltage scaling – voltage boosted up (down) for slower (faster) dies. The chips in slower/discarded bins move to faster bins

3.3.4.2 Adaptive Voltage Scaling

Circuit speed is a strong function of supply voltage [58]. Although increasing the supply voltage speeds up the circuit, it also worsens the power consumption. However, careful tuning of supply voltage can indeed improve the frequency while staying within the power budget. For example, the slower dies already consume low power due to high V_{TH} transistors. Therefore, supply voltage of these dies can be boosted up to improve the frequency. Similarly, the supply voltage of faster and power hungry dies can be scaled down to save power dissipation while affecting the speed minimally. In [58], the authors propose adaptive supply voltage technique to improve frequency and meet power specification. As shown in Fig. 3.13b, The slower dies from bin3 can be moved to nominal speed bin2 and faster dies from bin1 to bin2.

3.3.4.3 Programmable Designs

Several post-Silicon techniques have been proposed to tune the design and improve the robustness [96]. One such example is [96], where the authors describe a Process-Compensating Dynamic (PCD) circuit technique for register files. Unlike prior fixed-strength keeper techniques [102], the keeper strength is optimally programmed based on the respective die leakage. Figure 3.14 shows the PCD scheme with a digitally programmable three-bit keeper applied on an eight-way register file local bitline (LBL). Each of the three binary-weighted keepers with respective widths W, $2W$, and $4W$ can be activated or deactivated by asserting appropriate globally routed signals b[2:0]. A desired effective keeper width can be chosen among $[0, W, 2\,W, \ldots, 7\,W]$. The programmable keeper improves robustness and delay variation spread by restoring robustness of worst case leakage dies and improving performance of low-leakage dies.

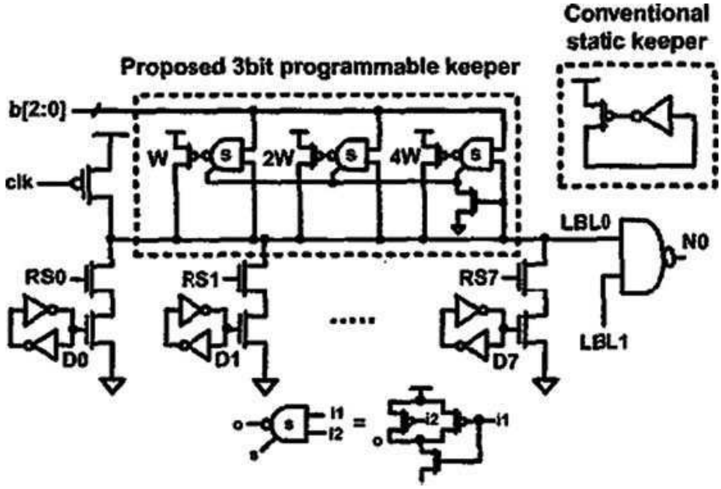


Fig. 3.14 Register file 8-way LBL with proposed PCD technique [96]

3.3.4.4 Sensors Based Design

Design time techniques are prone to errors due to operating PVT conditions [97–99]. Therefore, on-chip sensors can be used to estimate the extent of variations and apply right amount of corrective actions to avoid any possible failures. A variation-resilient circuit design technique is presented in [97] for maintaining parametric yield under inherent variation in process parameters. It utilizes on-chip phase locked loop (PLL) as a sensor to detect process, V_{DD}, and temperature (PVT) variations, or even temporal degradation stemming from negative bias temperature instability (NBTI) and uses adaptive body bias for correction. Several similar sensor-based adaptive design techniques can be found in literature. For example, [98] presents an on-chip wearout detection circuit whereas an adaptive multichip processor based on online NBTI and oxide breakdown detection sensors has been presented in [99]. Details of this approach can be found in following chapters.

3.3.4.5 RAZOR

RAZOR [65–69] uses dynamic detection and correction of circuit timing errors by using a shadow latch to tune processor supply voltage. This technique is quite effective in eliminating the timing margins due to process variation and environmental fluctuations. More details on RAZOR can be found in Chapter 7.

Note that the power management techniques, e.g., supply voltage scaling, power gating, multiple-V_{DD}, and V_{TH} designs further magnify the problems associated with process-induced variations. Power and process variation resilience are therefore conflicting design requirements and one comes at the cost of other. Meeting a desired power specification with certain degree of process tolerance is a stiff challenge and topic of further research.

3.4 Temporal Variability Management Techniques

In previous section, we discussed various pre-Si and post-Si adaptive techniques to overcome the impact of process-induced spatial variation. This section will summarize the recent techniques to tolerate temporal degradation (voltage, temperature, and NBTI). These techniques will be described in the following paragraphs.

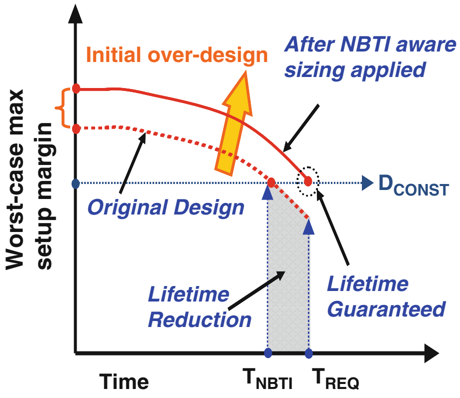
3.4.1 Voltage and Temperature Fluctuation Tolerance

Voltage fluctuation results from a combination of higher switching activity of the underlying circuit and weak power grid. This type of variation can be tolerated by inserting large decoupling capacitors in the power grid. The uneven temperature distribution within the chip is the outcome of excessive power consumption by the circuit. The power dissipates as heat and if the package is unable to sink the heat generated by the circuit, then it is manifested as elevated temperature. Since different parts of the chip experience different switching activity and power consumption, the temperature profile of the chip also varies accordingly over time. The temperature can be managed by reducing the power consumption of the die. For example, the operating frequency can be slowed down or the supply voltage can be scaled down. One can also throttle the heat by simultaneous control of voltage as well as frequency for cubic reduction in power consumption. Several other techniques have been proposed past like, logic shutdown, clock gating, functional block duplication, etc.

3.4.2 Gate/TR Sizing

One of the first approaches for reliability-aware design was based on an optimal gate sizing [27, 29]. For example, the method proposed in [27] uses modified Lagrangian Relaxation (LR) algorithm to compute optimal size of each gate under NBTI degradation. The basic idea of this approach is conceptually described in Fig. 3.15. As can be seen, the setup time margin of the design reduces with time due to NBTI, and after certain stress period (T_{NBTI}), the design may fail to meet the given timing constraint (D_{CONST}). To avoid such failures, the authors over-design (transistor upsizing considering signal probabilities) to ensure right functionality even after the required product lifetime T_{REQ}. The results from [27] reported an average area overhead of 8.7% for ISCAS benchmark circuits to ensure 3 year lifetime functionality at 70-nm node. An alternative method of transistor-level sizing was also proposed in [29]. In this approach, rather than sizing both the PMOS and NMOS at the same time, the authors applied different sizing factor for each of them. This method could effectively reduce unnecessary slacks in NMOS network (which is not affected by NBTI) and lower the overall area overhead. The results from [27] reported that the average overhead reduced by nearly 40% compared to [27].

Fig. 3.15 Transistor sizing for NBTI tolerance



3.4.3 Technology Mapping and Logic Synthesis

An alternative method is to consider NBTI at the technology mapping stage of the logic synthesis [19]. This approach is based on the fact that different standard cells have different NBTI sensitivity with respect to their input signal probabilities (probability that signal is logic “1”). With this knowledge, standard cell library is re-characterized with an additional signal probability dependency [20]. During logic synthesis, this new library is applied to properly consider the impact of

NBTI degradation. An average of 10% reduction in area overhead compared to the worst-case logic synthesis was reported using this method.

3.4.4 Guard-Banding

This is a very basic technique where the delay constraints are tightened with a fixed amount of delay guard-banding during sizing. Guard-band is selected as average delay degradations in these circuits for its lifetime years. Though guard-banding ignores the sensitivity of individual gates with respect to NBTI, delay degradation is weakly dependent on how the circuits were originally designed. Interestingly, it has been demonstrated in [33] that for a well-selected delay constraint; guard-banding indeed produces results comparable to the previous two approaches.

3.4.5 Self-Correction Using On-Chip Sensor Circuits

In practice, the guard-banding, sizing, and the synthesis methods introduced above require an accurate estimation of NBTI-induced performance degradation [24, 97, 103, 104]. This estimation may produce errors due to unpredictable temperature, activity (signal probability), or process parameter variations. One way to handle these problems is to employ an active on-chip reliability sensor [24, 103, 104]. Another approach proposed in [97] utilizes the on-chip phase locked loop (PLL) to perform reliability sensing. In these approaches, the actual amount of NBTI degradation can be used for further corrective actions. For example, in [97] the detected signal is efficiently transformed into an optimal body-bias signal to avoid possible timing failures in the target circuit. However, it is also essential to consider the additional design overhead-induced (e.g., increased area, power, and interconnections) by body biasing and the sensor circuits. It should be further noted that self-correcting design techniques lets one design circuits using nominal or optimistic conditions (leading to lower power dissipation). Corrective actions, if required, are taken based on sensing NBTI/parameter degradations.

3.5 Conclusions

Parameter variation is becoming an important issue with scaling of device geometries. In this chapter, we described various sources of variations and their impact on logic circuits. We also demonstrated that variation-aware or error-aware design is essential to stay within the power/performance envelop while meeting the yield requirement. We presented various variation insensitive circuit design techniques to address these issues.

References

1. Asenov A, Brown AR, Davies JH, Kaya S, Slavcheva G (Sept 2003) Simulation of intrinsic parameter fluctuations in decananometer and nanometer-scale MOSFETs. IEEE Trans Electron Devices 50(9):1837–1852
2. Hane M, Kawakami Y, Nakamura H, Yamada T, Kumagai K., Watanabe Y (2003) A new comprehensive SRAM soft error simulation based on 3D device simulation incorporating neutron nuclear reactions. In: Proceeding of simulation of semiconductor processes and devices, Boston, MA, pp 239–242
3. Nassif SR (2001) Modeling and analysis of manufacturing variations. In: Proceeding of custom integrated circuit conf., San Diego, CA, pp 223–228
4. Visweswariah C (2003) Death, taxes and failing chips. In: Proceeding of design automation conference, Anaheim, CA, pp 343–347
5. Borkar S, Karnik T, Narendra S, Tschanz J, Keshavarzi A, De V (2003) Parameter variation and impact on circuits and microarchitecture. In: Proceeding of design automation conference, Anaheim, CA, pp 338–342
6. Bhavnagarwala A, Tang X, Meindl JD (2001) The impact of intrinsic device fluctuations on CMOS SRAM cell stability. IEEE J Solid State Circuits 36:658–665
7. Tang X, De V, Meindl JD (1997) Intrinsic MOSFET parameter fluctuations due to random dopant placement. Trans VLSI syst 5:369–376
8. Raychowdhury A, Keshavarzi A (2008) Theory of multi-tube carbon nanotube transistors for high speed variation-tolerant circuits. In: Proceeding of device research conference, Santa Barbara, CA, pp 23–24
9. Nieuwoudt A, Massoud Y (2007) Assessing the implications of process variations on future carbon nanotube bundle interconnect solutions. In: Proceeding of international symposium on quality electronic design, San Francisco, California
10. Patil N, Deng J, Wong HSP, Mitra S (2007) Automated design of misaligned-carbon-nanotube-immune circuits. In: Proceeding of design automation conference, San Diego, California, pp 958–961
11. Zhang J, Patil N, Hazeghi A, Mitra S (2009) Carbon nanotube circuits in the presence of carbon nanotube density variations. In: Proceeding of design automation conference, San Francisco, California, pp 71–76
12. Bobba S et al (2009) Design of compact imperfection-immune CNFET layouts for standard-cell-based logic synthesis. In: Proceeding of Design, Automation & Test in Europe, Nice
13. Borkar S et al (2005) Statistical circuit design with carbon nanotubes. U.S. Patent Application 20070155065
14. Gunther SH, Binns F, Carmean DM, Hall JC (2001) Managing the impact of increasing microprocessor power consumption. Intel Tech J 5, (1):1–9
15. Deal BE, Sklar M, Grove AS, Snow EH (1967) Characteristics of the surface-state charge (Qss) of thermally oxidized silicon. J Electrochem Soc 114:266
16. Nicollian EH, Brews JR (1982) MOS physics and technology. Wiley, New York, NY
17. Blat CE, Nicollian EH, Poindexter EH (1991) Mechanism of negative bias temperature instability. J Appl Phys 69:1712
18. Li MF et al (2004) Dynamic bias-temperature instability in ultrathin SiO_2 and HfO_2 metal-oxide semiconductor field effect transistors and its impact on device lifetime. Jpn J Appl Phys 43:7807–7814, November
19. Kumar SV, Kim CH, Sapatnekar SS (2007) NBTI-aware synthesis of digital circuits. In: Proceedings of the ACM/IEEE design automation conference, San Diego, CA, pp 370–375
20. Kumar SV, Kim CH, Sapatnekar SS (2006) An analytical model for negative bias temperature instability. In: Proceedings of the IEEE/ACM international conference on computer-aided design, San Jose, CA, pp 493–496

21. Kumar SV, Kim CH, Sapatnekar SS (2006) Impact of NBTI on SRAM read stability and design for reliability. In: Proceedings of the international symposium on quality electronic design, San Jose, CA, pp 210–218
22. Kumar SV, Kim CH, Sapatnekar SS (2009) Adaptive techniques for overcoming performance degradation due to aging in digital circuits. In: Proceedings of the Asia-South Pacific design automation conference, Yokohama, pp 284–289
23. Kumar S, Kim CH, Sapatnekar S (2007) NBTI-aware synthesis of digital circuits. In: Proc. design automation conf., Dan Diego, CA, pp 370–375
24. Karl E, Singh P, Blaauw D, Sylvester D (Feb 2008) Compact in-situ sensors for monitoring negative-bias-temperature-instability effect and oxide degradation. IEEE international solid-state circuits conference, 2008 (ISSCC 2008). Digest of technical papers, San Francisco, CA, pp 410–623, 3–7
25. Wang W, Reddy V, Krishnan AT, Krishnan S, Cao Y (2007) An integrated modeling paradigm of circuit reliability for 65 nm CMOS technology. In: Proceeding of custom integrated circuits conference, San Jose, CA
26. Wang W, Wei Z, Yang S, Cao Y (2007) An efficient method to identify critical gates under circuit aging. In: Proceedings of the international conference on computer aided design (ICCAD) San Jose, CA
27. Kang K, Kufluoglu H, Alam MA, Roy K (2006) Efficient transistor-level sizing technique under temporal performance degradation due to NBTI. In: Proceeding of international conference on computer design, San Jose, CA, pp 216–221
28. Kunfluoglu H (2007) MOSFET degradation due to negative bias temperature instability (NBTI) and hot carrier injection (HCI) and its implications for reliability-aware VLSI design. PhD dissertation, Purdue University
29. Paul BC, Kang K, Kuflouglu H, Alam MA, Roy K (2006) Temporal performance degradation under NBTI: estimation and design for improved reliability of nanoscale circuits. In: Proc. design automation and test in Europe, Munich, pp 780–785
30. Paul BC, Kang K, Kufluoglu H, Alam MA, Roy K (2005) Impact of NBTI on the temporal performance degradation of digital circuits. IEEE Electron Device Lett 26(8):560–562
31. Kang K, Alam MA, Roy K (2007) Characterization of NBTI induced temporal performance degradation in nano-scale SRAM array using IDDQ. In: Proc. intl. test conference, Santa Clara, CA, pp 1–10
32. Kang K, Park SP, Roy K, Alam MA (2007) Estimation of statistical variation in temporal NBTI degradation and its impact in lifetime circuit performance. In: Proc. international conference on computer aided design, San Jose, CA, pp 730–734
33. Kang K, Gangwal S, Park SP, Roy K (2008) NBTI induced performance degradation in logic and memory circuits: how effectively can we approach a reliability solution?. In: Proc. Asia and South Pacific design automation conference, Seoul, pp 726–731
34. Jafar S, Kim YH, Narayanan V, Cabral C, Paruchuri V, Doris B, Stathis J, Callegari A, Chudzik M (2006) A comparative study of NBTI and PBTI (charge trapping) in SiO2/HfO2 stacks with FUSI, TiN, Re gates. In: Proceeding of VLSI circuits, Honolulu, HI, pp 23–25
35. Crupi F et al (Jun 2005) Positive bias temperature instability in nMOSFETs with ultra-thin Hf-silicate gate dielectrics. J Microelectron Eng 80:130–133
36. Ning TH, Cook PW, Dennard RH, Osburn CM, Schuster SE, Yu H (1979) 1 μm MOSFET VLSI technology: part IV-Hot electron design constraints. Trans Electron Devices 26: 346–353
37. Abramo A, Fiegna C, Venturi F (1995) Hot carrier effects in short MOSFETs at low applied voltages. In: Proc. intl. electron device meeting, Washington, DC, pp 301–304
38. Taur Y, Ning TH (1998) Fundamentals of modern VLSI devices. Cambridge University Press, New York, NY
39. JEP122-A (2002) Failure mechanisms and models for semiconductor devices. JEDEC Publication, JEDEC solid state technology association

40. Quddus MT, DeMassa TA, Sanchez JJ (2000) Unified model for Q(BD) prediction for thin gate oxide MOS devices with constant voltage and current stress. Microelectron Eng 51–52:357–372
41. MA Alam, Weir B, Silverman A (2002) A future of function or failure. IEEE Circuits Devices Mag 18:42–48
42. D Young, Christou A (1994) Failure mechanism models for electromigration. IEEE Trans Reliab 43:186–192
43. Boning D, Nassif S (2001) Models of process variations in device and interconnect. Design of high performance microprocessor circuits. Wiley, New York, NY.
44. Sirichotiyakul S et al (Apr 2002) Duet: an accurate leakage estimation and optimization tool for dual-Vt circuits. IEEE Trans VLSI Syst 10:79–90
45. Pant P, Roy R, Chatterjee A (2001) Dual-threshold voltage assignment with transistor sizing for low power CMOS circuits. IEEE Trans VLSI Syst 9:390–394
46. Wei L et al (1998) Design and optimization of low voltage high performance dual threshold CMOS circuits. In: Proceeding of design automation conference, San Francisco, CA, pp 489–494
47. Karnik T et al (2002) Total power optimization by simultaneous dual-Vt allocation and device sizing in high performance microprocessors. In: Proceeding of design automation conference, New Orleans, LA, pp 486–491
48. Nguyen D et al (2003) Minimization of dynamic and static power through joint assignment of threshold voltages and sizing optimization. In: Proceeding of international symposium on low-power electronics design, Seoul, pp 158–163
49. Srivastava A et al (2003) Simultaneous Vt selection and assignment for leakage optimization. In: Proceeding of international symposium on low-power electronics design, Seoul, pp 146–151
50. Sundarajan V, Parhi K (1999) Low power synthesis of dual threshold voltage CMOS VLSI circuits. In: Proceeding of international symposium on low-power electronics design, San Diego, CA, pp 139–144
51. Srivastava A, Sylvester D, Blauuw D, Agarwal A (2004) Statistical optimization of leakage power considering process variations using dual-V_{TH} and sizing. In: Proceeding of design automation conference, San Diego, CA, pp 773–778
52. Ketkar M et al (2000) Convex delay models for transistor sizing. In: Proceeding of design automation conference, Los Angeles, CA, pp 655–660
53. Singh J, Nookala V, Luo Z-Q, Sapatnekar S (2005) Robust gate sizing by geometric programming. In: Proceeding of DAC, Anaheim, CA, pp 315–320
54. Choi SH, Paul BC, Roy K (2004) Novel sizing algorithm for yield improvement under process variation in nanometer. In: Proceeding of design automation conf., San Diego, CA, pp 454–459
55. Chen C-P, Chu CCN, Wong DF (1999) Fast and exact simultaneous gate and wire sizing by Lagrangian relaxation. IEEE Trans Comput Aided Des, 18:1014–1025
56. Bai X, Visweswariah C, Strenski PN, Hathaway DJ (2002) Uncertainty-aware circuit optimization. In: Proceeding of design automation conf., New Orleans, LA, pp 58–63
57. Borkar S, Karnik T, De V (2004) Design and reliability challenges in nanometer technologies. In: Proceeding of design automation conference, San Diego, CA, pp 75–75
58. Borkar S, Karnik T, Narendra S, Tschanz J, Keshavarzi A, De V (2003) Parameter variations and impact on circuits and microarchitecture. DAC, Anaheim, CA, pp 338–342
59. Kumar SV, Kim CH, Sapatnekar SS (2006) Mathematically-assisted adaptive body bias (ABB) for temperature compensation in gigascale LSI systems. In: Proceedings of the Asia-South Pacific design automation conference, Yokohama, pp 559–564
60. Kumar SV, Kim CH, Sapatnekar SS (Mar 2008) Body bias voltage computations for process and temperature compensation. IEEE Trans VLSI Syst 16(3):249–262
61. Zhuo C, Blaauw D, Sylvester D (2008) Variation-aware gate sizing and clustering for post-silicon optimized circuits. ISLPED, Bangalore, pp 105–110

62. Kulkarni S, Sylvester D, Blaauw D (2006) A statistical framework for post-silicon tuning through body bias clustering. In: Proceeding of ICCAD, San Jose, CA, pp 39–46
63. Mani M, Singh A, Orshansky M (2006), Joint design-time and postsilicon minimization of parametric yield loss using adjustable robust optimization. In: Proceeding of ICCAD, San Jose, CA, pp 19–26
64. Khandelwal V, Srivastava A (2006) Variability-driven formulation for simultaneous gate sizing and post-silicon tunability allocation. In: Proceeding of ISPD, Austin, TA, pp 17–25
65. Ernst D, Kim NS, Das S, Pant S, Pham T, Rao R, Ziesler C, Blaauw D, Austin T, Mudge T (2003) Razor: a low-power pipeline based on circuit-level timing speculation. In: Proceeding of international symposium on microarchitecture, pp 7–18
66. Bowman KA, Tschanz JW, Nam Sung Kim Lee JC, Wilkerson CB, Lu S-LL, Karnik T, De VK (2008) Energy-efficient and metastability-immune timing-error detection and instruction-replay-based recovery circuits for dynamic-variation tolerance. In: Solid-state circuits conference (ISSCC 2008), San Francisco, CA, pp 402–623
67. Blaauw D, Kalaiselvan S, Lai K, Ma W-H, Pant S, Tokunaga C, Das S, Bull D (Feb 2008) RazorII: in-situ error detection and correction for PVT and SER tolerance. In: IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, CA
68. Austin T, Blaauw D, Mudge T, Flautner K (Mar 2004) Making typical silicon matter with Razor. IEEE Comput 37(3):57–65
69. Ernst D, Das S, Lee S, Blaauw D, Austin T, Mudge T, Nam Sung Kim, Flautner K (Nov-Dec 2004) Razor: circuit-level correction of timing errors for low-power operation. IEEE 24(6):10–20
70. Liang X, Wei G, Brooks D (Dec 2007) Process variation tolerant 3T1D based cache architectures. In: IEEE international symposium on microarchitecture, Chicago, IL
71. Liang X, Brooks D (Dec 2006) Mitigating the impact of process variations on processor register files and execution units. In: IEEE international symposium on microarchitecture, Orlando, FL
72. Ghosh V, Mahapatra D, Karakonstantis G, Roy K (Sep 2010) Low-voltage high-speed robust hybrid arithmetic units using adaptive clocking. IEEE Trans VLSI (accepted) 18:1301–1309
73. Mohapatra D, Karakonstantis G, Roy K (2007) Low-power process-variation tolerant arithmetic units using input-based elastic clocking. In: ISLPED, Portland, OR, pp 74–79
74. Tiwari A, Sarangi SR, Torrellas J (2007) Recycle: pipeline adaptation to tolerate process variation. In: Proceedings of the international symposium on computer architecture, San Diego, CA
75. Liang X, Wei G-Y, Brooks D (Jun 2008) ReVIVaL: a variation tolerant architecture using voltage interpolation and variable latency. In: Proceedings of the international symposium on computer architecture (ISCA-35), Beijing
76. Liang X, Brooks D, Wei G-Y (Feb 2008) A process-variation-tolerant floating-point unit with voltage interpolation and variable latency. In: IEEE international solid-state circuits conference, San Francisco, CA, pp 404–623
77. Ghosh S, Bhunia S, Roy K (2007) CRISTA: a new paradigm for low-power and robust circuit synthesis under parameter variations using critical path isolation. IEEE Trans Comput Aided Des
78. Shanbhag NR (2002) Reliable and energy-efficient digital signal processing. In: Proceeding of design automation conference, New Orleans, LA, pp 830–835
79. Kumar SV, Kashyap C, Sapatnekar SS (2008) A framework for block-based timing sensitivity analysis. In: Proceedings of the ACM/IEEE design automation conference, Anaheim, CA, pp 688–693
80. Blaauw D, Chopra K, Srivastava A, Scheffer L (2008) Statistical timing analysis: from basic principles to state of the art. IEEE Trans Comput Aided Des 27:589–607
81. Datta A, Bhunia S, Mukhopadhyay S, Banerjee N, Roy K (2005) Statistical modeling of pipeline delay and design of pipeline under process variation to enhance yield

in sub-100 nm technologies. In: Proceeding of design automation and test in Europe, pp 926–931

82. Orshansky M, Keutzer K (2002) A general probabilistic framework for worst case timing analysis. In: Design automation conference, New Orleans, LA, pp 556–561
83. Visweswariah C, Ravindran K, Kalafala K, Walker SG, Narayan S, Beece DK, Piaget J, Venkateswaran N, Hemmett JG (2006) First-order incremental block-based statistical timing analysis. IEEE Trans Comput Aided Des Integr Circ Syst 25: 2170–2180
84. Rao R, Srivastava A, Blaauw D, Sylvester D (2004) Statistical analysis of subthreshold leakage current for VLSI circuits. Trans VLSI syst 12:131–139
85. Zhang S, Wason V, Banerjee K (2004) A probabilistic framework to estimate full-chip subthreshold leakage power distribution considering within-die and die-to-die P-T-V variations. In: Proceeding of international symposium on low power electronics and design, Newport Beach, CA, pp 156–161
86. Rao R, Devgan A, Blaauw D, Sylvester D (2004) Parametric yield estimation considering leakage variability. In: Proceeding of design automation conference, San Diego, CA, pp 442–447
87. Agrawal A, Kang K, Roy K (2005) Accurate estimation and modeling of total chip leakage considering inter- & intra-die process variations. In: Proceeding of international conference on computer aided design, San Jose, CA, pp 736–741
88. Wang T-Y, Chen CC-P (Dec 2002) 3-D thermal-ADI: A linear-time chip level transient thermal simulator. IEEE Trans Comput Aided Des Integr Circ Syst, vol. 21, no. 12, pp 1434–1445
89. Su H, Liu F, Devgan A, Acar E, Nassif S (Aug 2003) Full chip estimation considering power, supply and temperature variations. In: Proceeding of international symposium low power electron. design, pp 78–83
90. Li P, Pileggi L, Asheghi M, Chandra R (2004) Efficient full-chip thermal modeling and analysis. In: Proceedings of international conference on computer aided design, Seoul, pp 319–326
91. Cheng Y, Raha P, Teng C, Rosenbaum E, Kang S (Aug 1998) ILLIADS-T: an electrothermal timing simulator for temperature-sensitive reliability diagnosis of CMOS VLSI chips. IEEE Trans Comput Aided Des Integr Circ Syst 17(8):668–681
92. Huang W, Stan MR, Skadron K, Sankaranarayanan K, Ghosh S (May 2006) HotSpot: a compact thermal modeling method for CMOS VLSI systems. IEEE Trans VLSI Syst 14(5):501–513
93. BPTM: Berkeley predictive technology model. http://www-device.eecs.berkeley.edu/∼ptm
94. Kang K, Paul BC, Roy K (2006) Statistical timing analysis using levelized covariance propagation considering systematic and random variations of process parameters. ACM Trans Des Autom Electron Syst 11:848–879
95. Datta A, Bhunia S, Mukhopadhyay S, Roy K (2005) A statistical approach to area-constrained yield enhancement for pipelined circuits under parameter variations. In: Proceeding of Asian test symposium, Kolkata, pp 170–175
96. Kim CH, Roy K, Hsu S, Alvandpour A, Krishnamurthy R, Borkhar S (2003) A process variation compensating technique for Sub-90 nm dynamic circuits. In: symposium on VLSI circuits, Kyoto
97. Kang K, Kim K, Roy K (2007) Variation resilient low-power circuit design methodology using on-chip phase locked loop. In: Proceeding of design automation conference, San Diego, CA, pp 934–939
98. Blome J, Feng S, Gupta S, Mahlke S (Dec. 2007) Self-calibrating online wearout detection. In: Proc. 40th intl. symposium on microarchitecture (MICRO), Chicago, IL, pp 109–120
99. Feng S, Gupta S, Mahlke S (2008) Olay: combat the signs of aging with introspective reliability management. The Workshop on quality-aware design (W-QUAD)

100. Ghosh S, Batra P, Kim K, Roy K (2007) Process-tolerant low-power adaptive pipeline under scaled-Vdd. In: Proceeding of custom integrated circuits conference, San Jose, CA, pp 733–736
101. Ghosh S, Choi JH, Ndai P, Roy K (2008) O^2C: occasional two-cycle operations for dynamic thermal management in high performance in-order microprocessors. In: Proceeding of international symposium on low power electronics and design, Bengaluru, pp 189–192
102. Krishnamurthy R, Alvandpour A, Balamurugan G, Shanbag N, Soumyanath K, Borkar S (2002) A 130 nm 6-GHz 256×32 bit Leakage-Tolerant Register File, IEEE J Solid State Circuits 37:624–632
103. Kim T, Persaud R, Kim CH (2007) Silicon odometer: an on-chip reliability monitor for measuring frequency degradation of digital circuits. In: Proceeding of VLSI circuit symposium, Kyoto, pp 122–123
104. Agarwal M, Paul BC, Zhang M, Mitra S (2007) Circuit failure prediction and its application to transistor aging. In: Proceeding of VLSI test symposium, Kyoto, pp 277–286

Chapter 4
Statistical Design of Integrated Circuits

Sachin S. Sapatnekar

Abstract The presence of process variations makes it imperative to depart from the traditional corner-based methodology and migrate to statistical design techniques. In this chapter, based on a set of variational models that capture correlated as well as uncorrelated variations, we present techniques for presilicon statistical timing and power analysis to determine the performance spread over a population of manufactured parts. In order to improve this spread, we discuss presilicon statistical optimization techniques that incorporate appropriate margins to enable improved manufacturing yield. At the post-silicon stage, we then present how a set of compact sensors may be used to predict the delay of a manufactured part, with known confidence, through a small set of measurements on the sensors: such data can then be used to drive adaptive post-silicon tuning approaches that are individualized to each manufactured part.

4.1 Introduction

As feature sizes have moved into tens of nanometers, it has become widely accepted that design tools must account for parameter variations during manufacturing. These considerations are important during both circuit analysis and optimization, in the presilicon as well as the post-silicon phases, and are essential to ensure circuit performance and manufacturing yield. These sources of variation can broadly be categorized into three classes:

- *Process variations* result from perturbations in the fabrication process, due to which the nominal values of parameters such as the effective channel length (L_{eff}), the oxide thickness (t_{ox}), the dopant concentration (N_{a}), the transistor width

S.S. Sapatnekar (✉)
Department of Electrical and Computer Engineering, University of Minnesota,
Minneapolis, MN, USA
e-mail: sachin@umn.edu

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_4,

(w), the interlayer dielectric (ILD) thickness (t_{ILD}), and the interconnect height and width (h_{int} and w_{int}, respectively).

- *Environmental variations* arise due to changes in the operating environment of the circuit, such as the temperature or variations in the supply voltage (V_{dd} and ground) levels or soft errors. There is a wide body of work on analysis techniques to determine environmental variations, both for thermal issues and voltage drop, and a reasonable volume on soft errors.
- *Aging variations* come about due to the degradation of the circuit during its operation in the field. These variations can result in changes in the threshold voltage over time, or catastrophic failures due to prolonged stress conditions.

All of these types of variations can result in changes in the timing and power characteristics of a circuit. Process variations, even random ones, are fully determined when the circuit is manufactured and do not change beyond that point. Therefore, a circuit that experiences large variations can be discarded after manufacturing test, at the cost of yield loss. An optimization process can target the presilicon maximization of yield over the entire population of die, or a post-silicon repair mechanism. On the other hand, environmental variations may appear, disappear, and reappear in various parts of the circuit during its lifetime. Since the circuit is required to work correctly at every single time point during its lifetime and over all operating conditions, these are typically worst-cased. Aging variations are deterministic phenomena that can be compensated for by adding margins at the presilicon, or by adaptation at the post-silicon phase.

For these reasons, process variations are a prime target for statistical design that attempts to optimize the circuit over a range of random variations, while environmental and aging variations are not. The move to statistical design is a significant shift in paradigm from the conventional approach of deterministic design. Unlike conventional static timing analysis (STA) which computes the delay of a circuit at a specific process corner, statistical static timing analysis (SSTA) provides a probability density function (PDF)[1] of the delay distribution of the circuit over all variations. Similarly, statistical power analysis targets the statistical distribution of the power dissipation of a circuit.

Process parameter variations can be classified into two categories: across-die (also known as inter-die) variations and within-die (or intra-die) variations. Across-die variations correspond to parameter fluctuations from one chip to another, while within-die variations are defined as the variations among different locations within a single die. Within-die variations of some parameters have been observed to be spatially correlated, i.e., the parameters of transistors or wires that are placed close to each other on a die are more likely to vary in a similar way than those of transistors or wires that are far away from each other. For example, among the process parameters for a transistor, the variations of channel length L_{eff} and transistor width W are seen to have such spatial correlation structure, while parameter variations such as

[1] Equivalently, its integral, the cumulative density function (CDF), may be provided.

the dopant concentration N_A and the oxide thickness T_{ox} are generally considered not to be spatially correlated.

If the only variations are across-die variations, as was the case in older technologies, then the approach of using process corners is very appropriate. In such a case, all variations on a die are similar, e.g., all transistor L_{eff} values may be increased or decreased by a consistent amount, so that a worst-case parameter value may be applied. However, with scaling, the role of within-die variations has increased significantly. Extending the same example, such variations imply that some L_{eff} values on a die may increase while others may decrease, and they may do so by inconsistent amounts. Therefore, worst-case corners are inappropriate for this scenario, and statistically based design has become important.

This chapter begins by overviewing models for process variations in Section 4.2. Next, we survey a prominent set of techniques for statistical timing and power analysis in Sections 4.3 and 4.4, respectively. Presilicon optimization methods are outlined in Section 4.5, and statistically based sensing techniques are described in Section 4.6.

4.2 Mathematical Models for Process Variations

4.2.1 Modeling Variations

In general, the intra-chip process variation δ can be decomposed into three parts: a deterministic global component, δ_{global}; a deterministic local component δ_{local}; and a random component ε [1]:

$$\delta = \delta_{\text{global}} + \delta_{\text{local}} + \varepsilon \tag{4.1}$$

The global component, δ_{global}, is location-dependent, and several models are available in the literature to incorporate various known deterministic effects. The local component, δ_{local}, is proximity-dependent and layout-specific. The random residue, ε, stands for the random intra-chip variation and is modeled as a random variable with a multivariate distribution ε to account for the spatial correlation of the intra-chip variation. It is common to assume that the underlying distribution is Gaussian, i.e., $\varepsilon \sim N(0, \Sigma)$, where Σ is the covariance matrix of the distribution. However, other distributions may also be used to model this variation. When the parameter variations are assumed to be uncorrelated, Σ is a diagonal matrix; spatial correlations are captured by the off-diagonal cross-covariance terms in a general Σ matrix. A fundamental property of covariance matrices says that Σ must be symmetric and positive semidefinite.

To model the intra-die spatial correlations of parameters, the die region may be partitioned into $n\text{row} \times n\text{col} = n$ grids. Since devices or wires close to each other are more likely to have similar characteristics than those placed far away, it is reasonable to assume perfect correlations among the devices (wires) in the same grid,

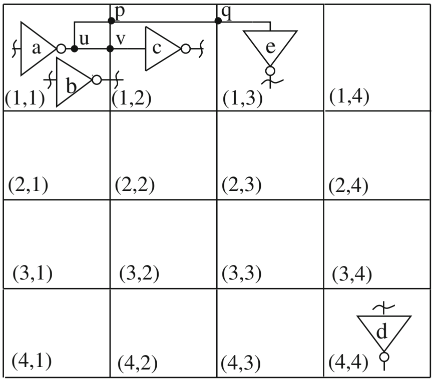


Fig. 4.1 Grid model for spatial correlations [2]

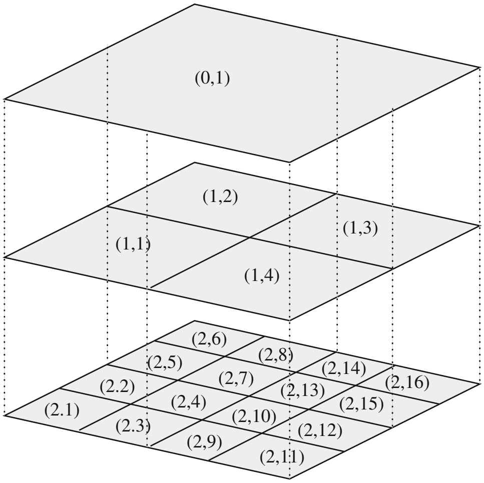
high correlations among those in close grids and low or zero correlations in far-away grids. For example, in Fig. 4.1, gates *a* and *b* (whose sizes are shown to be exaggeratedly large) are located in the same grid square, and it is assumed that their parameter variations (such as the variations of their gate length), are always identical. Gates *a* and *c* lie in neighboring grids, and their parameter variations are not identical but are highly correlated due to their spatial proximity. For example, when gate *a* has a larger than nominal gate length, it is highly probable that gate *c* will have a larger than nominal gate length, and less probable that it will have a smaller than nominal gate length. On the other hand, gates *a* and *d* are far away from each other, and their parameters are uncorrelated; for example, when gate *a* has a larger than nominal gate length, the gate length for *d* may be either larger or smaller than nominal.

Under this model, a parameter variation in a single grid at location (x, y) can be modeled using a single random variable $p(x, y)$. For each type of parameter, n random variables are needed, each representing the value of a parameter in one of the n grids.

In addition, it is reasonable to assume that correlation exists only among the same type of parameters in different grids and there is no correlation between different types of parameters. For example, the L_g values for transistors in a grid are correlated with those in nearby grids, but are uncorrelated with other parameters such as T_ox or W_int in any grid. For each type of parameter, an $n \times n$ covariance matrix, Σ, represents the spatial correlations of such a structure.

An alternative model for spatial correlations was proposed in [3, 4]. The chip area is divided into several regions using multiple quad-tree partitioning, where at level l, the die area is partitioned into $2^l \times 2^l$ squares; therefore, the uppermost level has just one region, while the lowermost level for a quad-tree of depth k has 4^k regions. A three-level tree is illustrated in Fig. 4.2. An independent random variable, $\Delta p_{i,r}$, is associated with each region (i, r) to represent the variations in parameter p in the region at level r. The total variation at the lowest level is then taken to be the sum of the variations of all squares that cover a region.

Fig. 4.2 The quadtree model for spatially correlated variations [3]



For example, in Fig. 4.2, in region (2,1), if p represents the effective gate length due to intra-die variations, $\Delta L_{\mathrm{eff}}(2,1)$, then

$$\Delta L_{\mathrm{eff}}(2,1) = \Delta L_{0,1} + \Delta L_{1,1} + \Delta L_{2,1} \tag{4.2}$$

In general, for region (i,j),

$$\Delta p(i,j) = \sum_{0<l<k,(l,r)\text{ covers }(i,j)} \Delta p_{l,r} \tag{4.3}$$

It can be shown rather easily that this is a special case of the model of Fig. 4.1, and has the advantage of having fewer characterization parameters. On the other hand, it shows marked edge effects that result in smaller correlations between adjacent cells if they fall across the edges of early levels of the quad-tree than those that do not.

Several approaches for characterizing spatial variations have been presented in the literature. The traditional approach is based on Pelgrom's model [5], which provides a closed-form structure for the variance of process parameters, and is widely used by analog designers to model device mismatch. In [6], a technique for fitting process data was presented, with a a guarantee that the resulting covariance matrix is positive definite. In [7], the notion behind Pelgrom's model is generalized using the idea of variograms to come up with a distance-based correlation model. An alternative radially symmetric spatial correlation model, based on hexagonal cells, was presented in [8].

4.2.2 Gaussian Models and Principal Components

When the underlying variations are Gaussian in nature, they are completely specified by a mean vector and a covariance matrix, Σ. However, working with correlated random variables involves considerable computation, and this can be reduced if the variables are orthogonalized into a basis set of independent random variables. Principal components analysis (PCA) techniques [9] convert a set of correlated random variables into a set of orthogonal uncorrelated variables in a transformed space; the PCA step can be performed as a preprocessing step for a design. As shown in [2], by performing this orthogonalization as a preprocessing step, once for each technology, the cost of SSTA can be significantly reduced. A variation on this theme is the idea of using the Kosambi-Karhunen-Loéve expansion [10], which allows correlations to be captured using a continuous, rather than a grid-based model and is useful for more fine-grained variations; indeed, PCA is sometimes referred to as the discrete osambi-Karhunen-Loéve transform.

Given a set of correlated random variables $\mathbf{X}$ with a covariance matrix Σ, the PCA method transforms the set $\mathbf{X}$ into a set of mutually orthogonal random variables, $\mathbf{P}$, such that each member of $\mathbf{P}$ has zero mean and unit variance. The elements of the set $\mathbf{P}$ are called principal components in PCA, and the size of $\mathbf{P}$ is no larger than the size of $\mathbf{X}$. Any variable $x_i \in \mathbf{X}$ can then be expressed in terms of the principal components $\mathbf{P}$ as follows:

$$x_i = \mu_i + \sigma_i \sum_{j=1}^{m} \sqrt{\lambda_i} \cdot v_{ij} \cdot p_j = \mu_i + \sum_{j=1}^{m} k_{ij} p_j \tag{4.4}$$

where p_{ij} is a principal component in set $\mathbf{P}$, λ_i is the ith eigenvalue of the covariance matrix Σ, v_{ij} is the ith element of the jth eigenvector of Σ, and σ_i and μ_i are, respectively, the mean and standard deviation of x_i. The term k_{ij} aggregates the terms that multiply p_j.

Since all of the principal components p_i that appear in Equation (4.4) are independent, the following properties ensue:

- The variance of d is given by

$$\sigma_{x_i}^2 = \sum_{i=1}^{m} k_{ij}^2 \tag{4.5}$$

- The covariance between x_i and any principal component p_j is given by

$$\mathrm{cov}(x_i, p_j) = k_{ij} \sigma_{p_j}^2 = k_{ij} \tag{4.6}$$

- For two random variables, x_i and x_l are given by

$$x_i = \mu_i + \sum_{j=1}^{m} k_{ij} p_j$$

$$x_l = \mu_l + \sum_{j=1}^{m} k_{lj} p_l$$

The covariance of x_i and x_l, $\mathrm{cov}(x_i, x_l)$ can be computed as

$$\mathrm{cov}(x_i, x_l) = \sum_{j=1}^{m} k_{ij} k_{lj} \tag{4.7}$$

> In other words, the number of multiplications is linear in the dimension of the space, since orthogonality of the principal components implies that the products of terms k_{ir} and k_{js} for $r \neq s$ need not be considered.

If we work with the original parameter space, the cost of computing the covariance is quadratic in the number of variables; instead, Equation (4.7) allows this to be computed in linear time. This forms the heart of the SSTA algorithm proposed in [2], and enables efficient SSTA.

4.2.3 Non-Gaussian Models and Independent Components

Non-Gaussian variations may be represented by a specific type of distribution in closed-form, or by a set of moments that characterize the distribution. These cases are indeed seen in practice: for example, the dopant density, N_{d}, can be modeled using a Poisson distribution. SSTA methods that work on non-Gaussians are generally based on moment-based formulations, and therefore, a starting point is in providing the moments of the process distribution.

Consider a process parameter represented by a random variable x_i: let us denote its kth moment by $m_k(x_i) = E[x_i^k]$. We consider three possible cases:

Case I: If the closed-form of the distribution of x_i is available and it is of a standard form (e.g., Poisson or uniform), then $m_k(x_i)\ \forall\ k$ can be derived from the standard mathematical tables of these distributions.

Case II: If the distribution is not in a standard form, then $m_k(x_i)\ \forall\ k$ may be derived from the moment generating function (MGF) if a continuous closed-form PDF of the parameter is known. If the PDF of x_i is the function $f_{x_i}(x_i)$, then its moment generating function $M(t)$ is given by

$$M(t) = E[\mathrm{e}^{tx_i}] = \int_{-\infty}^{\infty} \mathrm{e}^{tx_i} f_{x_i}(x_i) dx_i \tag{4.8}$$

The kth moment of x_i can then be calculated as the kth order derivative of $M(t)$ with respect to t, evaluated at $t = 0$. Thus, $m_k(x_i) = \frac{d^k M(t)}{dt^k}$ at $t = 0$.

Case III: If a continuous closed-form PDF cannot be determined for a parameter, the moments can still be evaluated from the process data files as:

$$m_k(x_i) = \sum_x x^k Pr(X_i = x) \tag{4.9}$$

where $\Pr(x_i = x)$ is the probability that the parameter x_i assumes a value x.

For variations that are not Gaussian-distributed, it is possible to use the independent component analysis method [11, 12] to orthogonalize the variables, enabling an SSTA solution that has a reasonable computational complexity [13].

4.3 Statistical Timing Analysis

The problem of SSTA is easily stated: given the underlying probability distributions of the process parameters, the goal of SSTA is to determine the probability distribution of the circuit delay. Most often, this task is divided into two parts: first, translating process variations into a gate-level probabilistic delay model, and then obtaining the circuit delay distribution.

Algorithms for SSTA can be classified according to various systems of taxonomy.

- *Path-based vs. block-based methods*: Path-based methods [3, 14] attempt to find the probability distribution of the delay on a path-by-path basis, and eventually performing a "max" operation to find the delay distribution of the circuit. If the number of paths to be considered is small, these methods can be effective, but in practice, the number of paths may be exponential in the number of gates. In contrast, block-based methods avoid path enumeration by performing a topological traversal, similar to that used by the critical path method (CPM), which processes each gate once when information about all of its inputs is known. While early approaches were predominantly path-based, state-of-the-art methods tend to operate in a block-based fashion.
- *Discrete vs. continuous PDFs*: SSTA methods can also be classified by their assumptions about the underlying probability distributions. Some approaches use discrete PDFs [15–17] while others are based on continuous PDFs; the latter class of techniques tend to dominate in the literature, although the former are capable of capturing a wider diversity of distributions, and may even directly use sample points from the process.
- *Gaussian vs. non-Gaussian models*: The class of continuous PDFs can be further subdivided into approaches that assume Gaussian (or normal) parameters, and those that permit more general non-Gaussian models.
- *Linear vs. nonlinear delay models*: Under small process perturbations, it is reasonable to assume that the change in gate delays follows a linear trend. However, as these perturbations grow larger, a nonlinear model may be necessary. Depending on which of these is chosen as the underlying model, the corresponding algorithm can incur smaller or larger computational costs.

The basic Monte Carlo method is probably the simplest method for performing statistical timing analysis. Given an arbitrary delay distribution, the method generates sample points and runs a static timing analyzer at each such point, and aggregates the results to find the delay distribution. The advantages of this method lie in its ease of implementation and its generality in being able to handle the complexities of variations and a wider range of delay models. For example, spatial correlations are easily incorporated, since all that is required is the generation of a sample point on a correlated distribution. Such a method is very compatible with the data brought in from the fab line, which are essentially in the form of sample points for the simulation. Its major disadvantage can be its extremely large runtimes. Recent work on SSTA has moved towards more clever and computationally efficient implementations [18–20]. Our discussion will largely focus on the faster and more widely used block-based SSTA methods that seek closed-form expressions for the delay at the output of each gate.

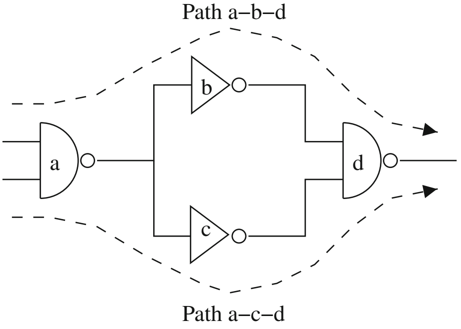


Fig. 4.3 An example to illustrate structural correlations in a circuit

In addition to accounting for randomness, including spatial correlations, SSTA algorithms must also consider the effects of correlations between delay variables due to the structure of the circuit. Consider the reconvergent fanout structure shown in Fig. 4.3. The circuit has two paths, a-b-d and a-c-d. The circuit delay is the maximum of the delays of these two paths, and these are correlated since the delays of a and d contribute to both paths.

4.3.1 Modeling Gate/Interconnect Delay PDF's

The variations in the process parameters translate into variations in the gate delays that can be represented as PDFs. Before we introduce how the distributions of gate and interconnect delays will be modeled, let us first consider an arbitrary function $d = f(\mathbf{P})$ that is assumed to be a function on a set of parameters $\mathbf{P}$, where each $p_i \in \mathbf{P}$ is a random variable with a known PDF. We can approximate d using a Taylor series expansion:

$$d = d_0 + \sum_{\forall \text{parameters}\, p_i} \left[\frac{\partial f}{\partial p_i} \right]_0 \Delta p_i + \sum_{\forall \text{parameters}\, p_i} \left[\frac{\partial^2 f}{\partial p_i^2} \right]_0 \Delta p_i^2 + \cdots \tag{4.10}$$

where d_0 is the nominal value of d calculated at the nominal values of parameters in the set **P**, $\left[\frac{\partial f}{\partial p_i}\right]_0$ is computed at the nominal values of and p_i, and $\Delta p_i = p_i - \mu_{p_i}$ is a zero-mean random variable. This delay expression is general enough to handle the effects of input slews and output loads; for details, see [21].

If all of the parameter variations can be modeled by Gaussian distributions, i.e., $p_i \sim N(\mu_{p_i}, \sigma_{p_i})$, then clearly $\Delta p_i \sim N(0, \sigma_{p_i})$. If a first-order Taylor series approximation is used in Equation (4.10) by neglecting quadratic and higher order terms, then d is a linear combination of Gaussians and is therefore Gaussian. Its mean μ_d and variance σ_d^2 are

$$\mu_d = d_0 \tag{4.11}$$

$$\sigma_d^2 = \sum_{\forall i} \left[\frac{\partial f}{\partial p_i} \right]_0^2 \sigma_{p_i}^2 + 2 \sum_{\forall i \neq j} \text{cov}(p_i, p_j) \tag{4.12}$$

where $\text{cov}(p_i, p_j)$ is the covariance of p_i and p_j.

In cases where the variations are larger than can be accurately addressed by a linear model, then higher-order terms of the expansion should be maintained. Most such nonlinear models in the literature (e.g., [22–24]) find it sufficient to consider the linear and quadratic terms in the Taylor expansion.

4.3.2 Algorithms for SSTA

4.3.2.1 Early Methods

Early work in this area spawned several methods that ignored the spatial correlation component, but laid the foundation for later approaches that overcame this limitation. Prominent among these was the work by Berkelaar in [25], [26], which presented a precise method for statistical static timing analysis that could successfully process large benchmarks circuits under probabilistic delay models. In the spirit of static timing analysis, this approach is purely topological and ignores the Boolean structure of the circuit. The underlying delay model assumes that each gate has a delay described by a Gaussian PDF, and observed that the essential operations in timing analysis can be distilled into two types:

SUM: A gate is processed when the arrival times of all inputs are known, at which time the candidate delay values at the output are computed using the “sum” operation that adds the delay at each input with the input-to-output pin delay.
MAX: The arrival time at the gate output is determined once these candidate delays have been found, and the “max” operation is applied to determine the maximum arrival time at the output.

The key to SSTA is to perform these two operations on operands that correspond to PDFs, rather than deterministic numbers as is the case for STA. Note that, as in STA, the SUM and MAX operators incorporate clock arrival times as well as signal arrival times.

Berkelaar's approach maintains an invariant that expresses all arrival times as Gaussians. As a consequence, since the gate delays are Gaussian, the "sum" operation is merely an addition of Gaussians, which is well known to be a Gaussian.

The computation of the "max" function, however, poses greater problems. The candidate delays are all Gaussian, so that this function must find the maximum of Gaussians. In general, the maximum of two Gaussians is *not* a Gaussian, but can be approximated as one. Intuitively, this can be justified by seeing that if a and b are Gaussian random variables, then

- if $a \gg b$, then $\max(a,b) = a$ is a Gaussian
- if $a = b$, then $\max(a,b) = a = b$ is a Gaussian

It was suggested in [25] that a statistical sampling approach could be used to approximate the mean and variance of the distribution; alternatively, this information could be embedded in look-up tables. In later work in [26], a precise closed-form approximation for the mean and variance, based on [27], was utilized.

4.3.2.2 Incorporating Spatial Correlations

In cases where significant spatial correlations exist, it is important to take them into account. Figure 4.4 shows a comparison of the PDF yielded by an SSTA technique that is unaware of spatial correlations, as compared with a Monte Carlo simulation that incorporates these spatial correlations, and clearly shows a large difference. This motivates the need for developing methods that can handle these dependencies.

Early approaches to spatial correlation did not scale to large circuits. The work in [28] extended the idea of [25] to handle intra-gate spatial correlations, while assuming zero correlation between gates. A notable feature of this work was the use of an approximation technique from [29] that provides a closed-form formula to approximate the maximum of two correlated Gaussian random variables as a Gaussian.

Under normality assumptions, the approach in [2, 21] leverages the decomposition of correlated variations into principal components, as described in Section 4.2.2, to convert a set of correlated random variables into a set of uncorrelated variables in a transformed space. As mentioned earlier, the PCA step is to be performed once for each technology as a precharacterization. The worst-case complexity of the method in [2, 21] is n times the complexity of CPM, where n is the number of squares in the correlation grid (see Fig. 4.1). The overall CPU times for this method have been shown to be low, and the method yields high accuracy results.

This parameterized approach to SSTA propagates a canonical form (a term popularized in [30]) of the delay PDF, typically including the nominal value, a set

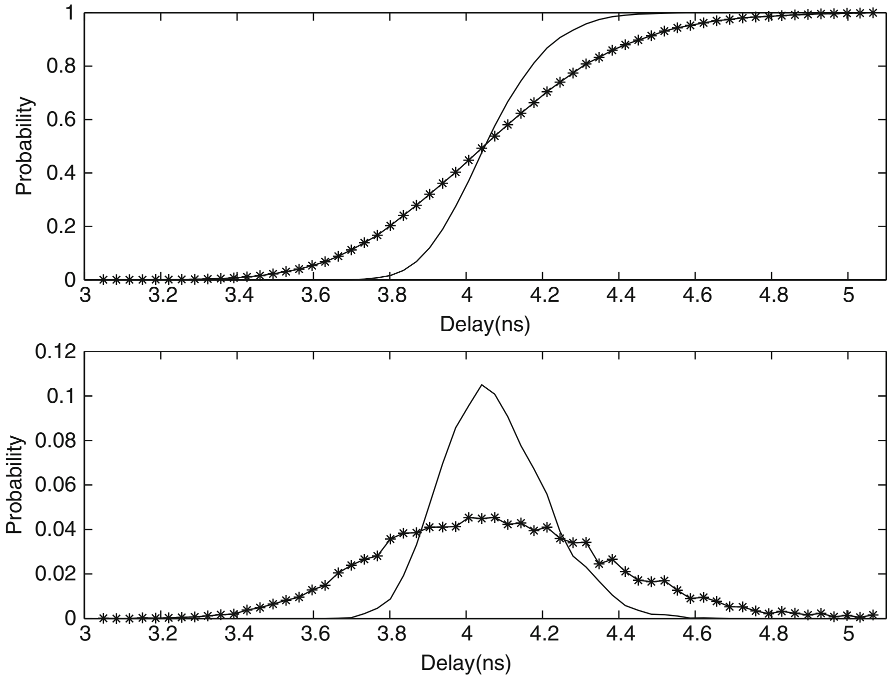


Fig. 4.4 A comparison of the results of SSTA when the random variables are spatially correlated. The line on which points are marked with stars represents the accurate results obtained by a lengthy Monte Carlo simulation, and the the solid curve shows the results when spatial correlations are entirely ignored. The upper plot shows the CDFs, and the lower plot, the PDFs [2]

of normalized underlying independent sources of variation. For spatially correlated variations, these sources correspond to the principal components (PCs) [2], computed by applying PCA to the underlying covariance matrix of the correlated variations; uncorrelated variations are typically captured by a single independent random variable.

If the process parameters are Gaussian-distributed, then the m PCs affect the statistical distribution of both the original circuit and the test structures on the same chip, and the canonical form for the delay d is represented as

$$d = \mu + \sum_{i=1}^{m} a_i p_i + R = \mu + \mathbf{a}^{\mathrm{T}}\mathbf{p} + R \tag{4.13}$$

where μ is the mean of the delay distribution. The value of μ is also an approximation of its nominal value.[2] The random variable p_i corresponds to the ith principal

[2] The nominal value of the delay of the circuit is the delay value when no parameter variations are present. This can be computed exactly by a conventional static timing analysis with all parameters

component, and is normally distributed, with zero mean and unit variance; note that p_i and p_j for $i \neq j$ are uncorrelated by definition, stemming from a property of PCA. The parameter a_i is the first order coefficient of the delay with respect to p_i. Finally, R corresponds to a variable that captures the effects of all the spatially uncorrelated variations. It is a placeholder to indicate the additional variations of the delay caused by the spatially uncorrelated variations, and cannot be regarded as a principal component.

Equation (4.13) is general enough to incorporate both inter-die and intra-die variations. It is well known that, for a spatially correlated parameter, the inter-die variation can be taken into account by adding a value σ^2_{inter}, the variance of inter-die parameter variation, to all entries of the covariance matrix of the intra-die variation of that parameter before performing PCA. The uncorrelated component R accounts for contributions from both the inter-die and intra-die variations. Systematic variations affect only the nominal values and the PC coefficients in SSTA. Therefore, they can be accounted for by determining the shifted nominal values and sensitivities prior to SSTA, and computing the nominal values and PC coefficients in SSTA based on these shifted values.

The work in [2] uses this canonical form, along with the properties of such a principal components-based representation (as described in Equations (4.5) through (4.7) to perform SSTA under the general spatial correlation model of Fig. 4.1.

The fundamental process parameters are assumed to be in the form of correlated Gaussians, so that the delay given by Equation (4.10) is a weighted sum of Gaussians, which is Gaussian.

As in the work of Berkelaar, this method maintains the invariant that all arrival times are approximated as Gaussians, although in this case the Gaussians are correlated and are represented in terms of their principal components. Since the delays are considered as correlated Gaussians, the sum and max operations that underlie this block-based CPM-like traversal must yield Gaussians in the form of principal components.

We will first consider the case where R in (Equation 4.13) is zero. The computation of the distribution of the sum function, $d_{\text{sum}} = \sum_{i=1}^{n} d_i$, is simple. Since this function is a linear combination of normally distributed random variables, d_{sum} is a normal distribution whose mean, $\mu_{d\text{sum}}$, and variance, $\sigma^2_{d\text{sum}}$, are given by

$$\mu_{d_{\text{sum}}} = \sum_{i=1}^{n} d_i^0 \tag{4.14}$$

$$\sigma^2_{d_{\text{sum}}} = \sum_{j=1}^{m} \sum_{i=1}^{n} k_{ij}^2 \tag{4.15}$$

at their nominal values. However, because of the approximation of the max operation in the statistical timer, the mean value computed from the topological traversal is more compatible with the rest of the canonical form.

where d_i is written in terms of its normalized principal components as $d_i^0 + \sum_{j=1}^{m} k_{ij}p_j$.

Strictly speaking, the max function of n normally distributed random variables, $d_{\max} = \max(d_1, \cdots, d_n)$, is not Gaussian; however, as before, it is approximated as one. The approximation here is in the form of a correlated Gaussian, and the procedure in [29] is employed. The result is characterized in terms of its principal components, so that it is enough to find the mean of the max function and the coeficients associated with the principal components.

Although the above exposition has focused on handling spatially correlated variables, it is equally easy to incorporate uncorrelated terms in this framework. Only spatially correlated variables are decomposed into principal components, and any uncorrelated variables are incorporated into the uncorrelated component, R, of (Equation 4.13); during the sum and max operations, the uncorrelated components of the operands are consolidated into a single uncorrelated component of the canonical form of the result. For a detailed description of the sum and max operations, the reader is referred to [21].

The utility of using principal components is twofold:

- As described earlier, it implies that covariance calculations between paths are of linear complexity in the number of variables, obviating the need for the expensive pair-wise delay computation methods used in other methods.
- In the absence of the random component, R, in (Equation 4.13), structural correlations due to reconvergent fanouts (see Fig. 4.3) are automatically accounted for, since all the requisite information required to model these correlations is embedded in the principal components. When R is considered, the structural components associated with R are lumped together and individual variational information is lost, leading to a slight degradation of accuracy. However, heuristic methods may be used to limit this degradation.

The overall flow of the algorithm is shown in Fig. 4.5. To further speed up the process, several techniques may be used:

Input: Process parameter variations
Output: Distribution of circuit delay

1. Partition the chip into $n = nrow \times ncol$ grids , each modeled by spatially correlated variables.
2. For each type of parameter, determine the n jointly normally distributed random variables and the corresponding covariance matrix.
3. Perform an orthogonal transformation to represent each random variable with a set of principal components.
4. For each gate and net connection, model their delays as linear combinations of the principal components generated in step 3.
5. Using "sum" and "max" functions on Gaussian random variables, perform a CPM-like traversal on the graph to find the distribution of the statistical longest path. This distribution achieved is the circuit delay distribution.

Fig. 4.5 Overall flow of the PCA-based statistical timing analysis method

1. Before running the statistical timing analyzer, one run of deterministic STA is performed to determine loose bounds on the best-case and worst-case delays for all paths. As in [31], any path whose worst-case delay is less than the best-case delay of the longest path will never be critical, and edges that lie only on such paths can safely be removed.
2. During the "max" operation of statistical STA, if the value of mean $+ 3 \cdot \sigma$ of one path has a lower delay than the value of mean $- 3 \cdot \sigma$ of another path, the max function can be calculated by ignoring the path with lower delay.

For the non-Gaussian case [13], the linear canonical form is similar to (Equation 4.13):

$$d = \mu + \mathbf{b}^{\mathrm{T}}\mathbf{x} + \mathbf{c}^{\mathrm{T}}\mathbf{y} + e.z \tag{4.16}$$

where d is the random variable corresponding to a gate delay or an arrival time at the input port of a gate. The vector $\mathbf{x}$ corresponds to the non-Gaussian independent components, obtained from applying ICA to the non-Gaussian process parameter set, and $\mathbf{b}$ is the vector of first-order sensitivities of the delay with respect to these independent components. The Gaussian random variables are orthogonalized using PCA into the principal component vector, $\mathbf{y}$, and $\mathbf{c}$ is the corresponding linear sensitivity vector. Finally, z is the uncorrelated parameter which may be a Gaussian or a non-Gaussian random variable, e is the sensitivity with respect this. We assume statistical independence between the Gaussian and non-Gaussian parameters: this is a reasonable assumption as parameters with dissimilar distributions are likely to represent different types of variables and are unlikely to be correlated.

The work in [13] presents an approach that translates the moments of the process parameters to the moments of the principal and independent components in a precharacterization step that is performed once for each technology. Next, a moment-based scheme is used to propagate the moments through the circuit, using a moment-matching scheme similar to the APEX algorithm [32]. The sum and max operations are performed on the canonical form to provide a result in canonical form, with moment-matching operations being used to drive the engine that generates the canonical form.

4.4 Statistical Power Analysis

The power dissipation of a circuit consists of the dynamic power, the short-circuit power, and the leakage power. Of these, the leakage power is increasing drastically with technology scaling, and has already become a substantial contributor to the total chip power dissipation. Consequently, it is important to accurately estimate leakage currents so that they can be accounted for during design, and so that it is possible to effectively optimize the total power consumption of a chip.

The major components of leakage in current CMOS technologies are due to subthreshold leakage and gate tunneling leakage. For a gate oxide thickness, T_{ox}, of over 20Å, the gate tunneling leakage current, I_{gate}, is typically very small, while the subthreshold leakage, I_{sub}, dominates other types of leakage in circuit. For this reason, early work on leakage focused its attention on subthreshold leakage. However, the gate tunneling leakage is exponentially dependent on gate oxide thickness, e.g., a reduction in T_{ox} of 2Å will result in an order of magnitude increase in I_{gate}. While high-K dielectrics provide some relief, the long-term trends indicate that gate leakage is an important factor. Unlike dynamic and short-circuit power, which are relatively insensitive to process variations, the circuit leakage can change significantly due to changes in parameters such as the transistor effective gate length and the gate oxide thickness. Therefore, statistical power analysis essentially equates to statistical leakage analysis.

4.4.1 Problem Description

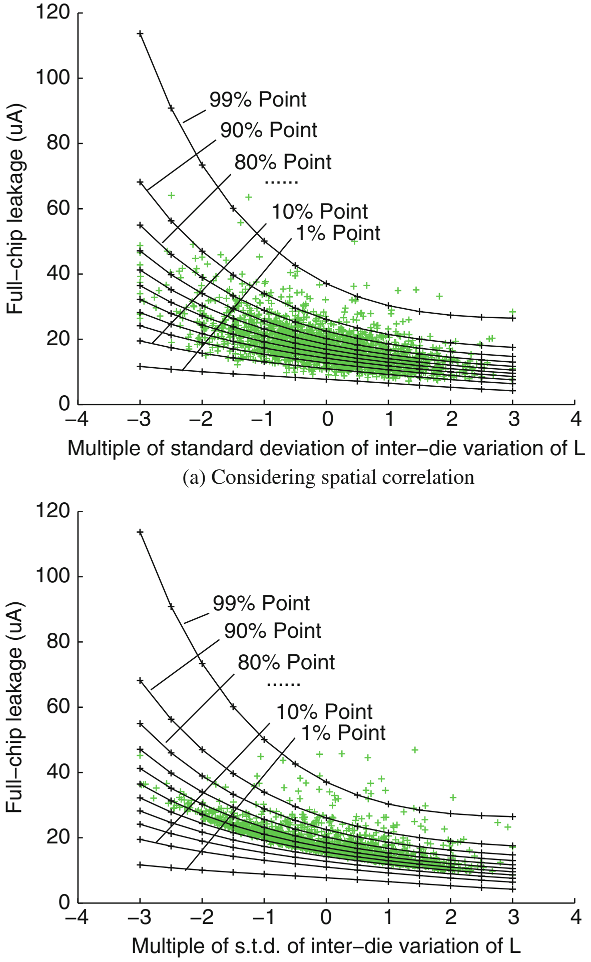
The total leakage power consumption of a circuit is input-pattern-dependent, i.e., the value differs as the input signal to the circuit changes, because the leakage power consumption, due to subthreshold and gate tunneling leakage, of a gate depends on the input vector state at the gate. As illustrated in [33], the dependency of leakage on process variations is more significant than on input vector states. Therefore, it is sufficient to predict the effects of process variations on total circuit leakage by studying the variation of average leakage current for all possible input patterns to the circuit. However, it is impractical to estimate the average leakage by simulating the circuit at all input patterns, and thus an input pattern-independent approach is more desirable.

In switching power estimation, probabilistic approaches [34] have been used for this purpose. The work of [33] proposed a similar approach that computes the average leakage current of each gate and estimates the total average circuit leakage as a sum of the average leakage currents of all gates:

$$I_{\mathrm{tot}}^{\mathrm{avg}} = \sum_{k=1}^{N_{\mathrm{g}}} I_{\mathrm{leak},k}^{\mathrm{avg}} = \sum_{k=1}^{N_{\mathrm{g}}} \sum_{\forall \mathrm{vec}_{i,k}} \mathrm{Prob}(\mathrm{vec}_{i,k}) \cdot I_{\mathrm{leak},k}(\mathrm{vec}_{i,k}) \tag{4.17}$$

where N_{g} is the total number of gates in the circuit, $I_{\mathrm{leak},k}^{\mathrm{avg}}$ is the average leakage current of the kth gate, $\mathrm{vec}_{i,k}$ is the ith input vector at the kth gate, $\mathrm{Prob}(\mathrm{vec}_{i,k})$ is the probability of occurrence of $\mathrm{vec}_{i,k}$, and $I_{\mathrm{leak},k}(\mathrm{vec}_{i,k})$ is the leakage current of the kth gate when the gate input vector is $\mathrm{vec}_{i,k}$.

In our discussion, we consider the variations in the transistor gate length L_{eff} and gate oxide thickness T_{ox}, since I_{sub} and I_{gate} are most sensitive to these parameters [35, 36]. To reflect reality, we model spatial correlations in transistor gate length, while the gate oxide thickness values for different gates are taken to be uncorrelated. Note that although only transistor gate length and gate oxide thickness are considered in this work, the framework is general enough to consider effects of any other types of process variations such as the channel dopant variation N_{d}.



(a) Considering spatial correlation

(b) Ignoring spatial correlation

Fig. 4.6 Comparison of scatter plots of full-chip leakage of circuit c432 considering and ignoring spatial correlation

In performing this computation, it is extremely important to consider the impact of spatial correlations. While random variations tend to cancel themselves out, spatially correlated variations magnify the extent of the variation. This difference can be visualized in Fig. 4.6, which shows the scatter plots for c432 for 2000 samples of full-chip leakage current generated by Monte Carlo simulations, with and without consideration of spatial correlations of L_{eff}. The x-axis marks the multiples of the standard deviation value of $\Delta L_{\mathrm{eff}}^{\mathrm{inter}}$, inter-die variations of effective gate length, ranging from −3 to +3, since a Gaussian distribution is assumed. The y-axis are the values of total circuit leakage current. Therefore, at each specific value of $\Delta L_{\mathrm{eff}}^{\mathrm{inter}}$, the scatter points list the various sampled values of total circuit leakage current due to variations in T_{ox} and intra-die variation of L_{eff}. The plots also show a set of contour lines that correspond to, with the effect of spatial correlation taken into account, a set of percentage points of the cumulative density function (CDF) of total circuit leakage current at different values of $\Delta L_{\text{eff}}^{\text{inter}}$. In Fig. 4.6a, where spatial correlations are considered, nearly all points generated from Monte Carlo simulation fall between the contours of the 1 and 99% lines. However, in Fig. 4.6b, where spatial correlations are ignored, the spread is much tighter in general: the average value of 90% point of full-chip leakage, with spatial correlation considered, is 1.5 times larger than that without for $\Delta L_{\text{eff}}^{\text{inter}} \leq -1\sigma$; the same ratio is 1.1 times larger otherwise. Looking at the same numbers in a different way, in Fig. 4.6b, all points are contained between the 30 and 80% contours if $\Delta L_{\text{eff}}^{\text{inter}} \leq -1\sigma$. In this range, I_{sub} is greater than I_{gate} by one order of magnitude on average, and thus the variation of L_{eff} can have a large effect on the total leakage as I_{sub} is exponentially dependent on L_{eff}. Consequently, ignoring spatial correlation results in a substantial underestimation of the standard deviation, and thus the worst-case full-chip leakage. For $\Delta L_{\text{eff}}^{\text{inter}} > -1\sigma$, I_{sub} decreases to a value comparable to I_{gate} and L_{eff} has a relatively weak effect on the variation of total leakage. In this range, the number of points of larger leakage values is similar to that when spatial correlation is considered. However, a large number of remaining points show smaller variations and are within the 20 and 90% contours, due to the same reasoning given above for $\Delta L_{\text{eff}}^{\text{inter}} \leq -1\sigma$.

4.4.2 Computing the Distribution of the Full-Chip Leakage Current

The distribution of $I_{\text{tot}}^{\text{avg}}$ can be calculated in two steps. First, given the probability of each input pattern vector to a gate, $\text{vec}_{i,k}$, we can compute the leakage of the gate as a weighted sum over all possible vectors. Second, this quantity can be summed up over all gates to obtain the total leakage. In other words,

$$I_{\text{tot}}^{\text{avg}} = \sum_{k=1}^{N_g} \sum_{\forall \text{vec}_{i,k}} \text{Prob}(\text{vec}_{i,k}) \cdot \big(I_{\text{sub},k}(\text{vec}_{i,k}) + I_{\text{gate},k}(\text{vec}_{i,k})\big) \tag{4.18}$$

where $I_{\text{leak},k}$ under vector $(\text{vec}_{i,k})$ is written as the sum of the subthreshold leakage, $I_{\text{sub},k}(\text{vec}_{i,k})$, and the gate leakage, $I_{\text{gate},k}(\text{vec}_{i,k})$, for gate k.

The commonly used model for subthreshold leakage current through a transistor expresses this current as [35]:

$$I_{\text{sub}} = I_0 \mathrm{e}^{(V_{\text{gs}}-V_{\text{th}})/n_{\text{s}}V_T}(1 - \mathrm{e}^{-V_{\text{ds}}/V_T}) \tag{4.19}$$

Here, $I_0 = \mu_0 C_{\text{ox}}(W_{\text{eff}}/L_{\text{eff}})V_T^2\mathrm{e}^{1.8}$, where μ_0 is zero bias electron mobility, C_{ox} is the gate oxide capacitance, W_{eff} and L_{eff} are the effective transistor width and length, respectively, V_{gs} and V_{ds} are gate-to-source voltage and drain-to-source voltage, respectively, n_{s} is the subthreshold slope coefficient, $V_T = kT/q$ is the thermal voltage, where k is Boltzman constant, T is the operating temperature in Kelvin (K), q is charge on an electron, and V_{th} is the subthreshold voltage.

It is observed that V_{th} is most sensitive to gate oxide thickness T_{ox} and effective transistor gate length L_{eff} due to short-channel effects [35]. Due to the exponential dependency of I_{sub} on V_{th}, a small change on L_{eff} or T_{ox} will have a substantial effect on I_{sub}. From this intuition, we estimate the subthreshold leakage current per transistor width by developing an empirical model through curve-fitting, similarly to [36, 37]:

$$I_{\mathrm{sub}} = c \times \mathrm{e}^{a_1 + a_2 L_{\mathrm{eff}} + a_3 L_{\mathrm{eff}}^2 + a_4 T_{\mathrm{ox}}^{-1} + a_5 T_{\mathrm{ox}}} \tag{4.20}$$

where c and the a_i terms are the fitting coefficients. To quantify the empirical model, the values of I_{sub} achieved from expression (Equation 4.20) are compared with those through SPICE simulations over a ranged values of T_{ox} and L_{eff}.

Under process perturbations, I_{sub} can be well approximated by expanding its exponent U using a first-order Taylor expansion at the nominal values of the process parameters:

$$I_{\mathrm{sub}} = c \times \mathrm{e}^{U_0 + \beta_1 \cdot \Delta L_{\mathrm{eff}} + \beta_2 \cdot \Delta T_{\mathrm{ox}}} \tag{4.21}$$

where U^0 is the nominal value of the exponent U, β_0 and β_1 are the derivatives of U to L_{eff} and T_{ox} evaluated at their nominal values, respectively, and ΔL_{eff} and ΔT_{ox} are random variables standing for the variations in the process parameters L_{eff} and T_{ox}, respectively.

Expression (Equation 4.21) for I_{sub} can also be written[3] as $\mathrm{e}^{\ln(c) + U_0 + \beta_1 \cdot \Delta L_{\mathrm{eff}} + \beta_2 \cdot \Delta T_{\mathrm{ox}}}$. Since ΔL_{eff} and ΔT_{ox} are assumed to be Gaussian-distributed, I_{sub} is seen as an exponential function of a Gaussian random variable, with mean $\ln(c) + U_0$ and standard deviation $\sqrt{\beta_1^2 \sigma_{L_{\mathrm{eff}}}^2 + \beta_2^2 \sigma_{T_{\mathrm{ox}}}^2}$, where $\sigma_{L_{\mathrm{eff}}}$ and $\sigma_{T_{\mathrm{ox}}}$ are standard deviations of ΔL_{eff} and ΔT_{ox}, respectively.

In general, if x is a Gaussian random variable, then $z = \mathrm{e}^x$ is a *lognormal random variable*. From (Equation 4.21), it is obvious that I_{sub} can be approximated as a lognormally distributed random variable whose probability density function can be characterized using the values of c, U_0, and β_i's.

Since subthreshold leakage current has a well-known input state dependency due to the stack effect [38], the PDFs of subthreshold leakage currents must be characterized for all possible input states for each type of gate in the library, for which the same approach described in this section can be applied. Once the library is characterized, a simple look-up table (LUT) can then be used to retrieve the corresponding model characterized given the gate type and input vector state at a gate.

[3] To consider the effect of varying N_{d} on I_{sub}, equation (4.21) can be adapted by adding an additional term for ΔN_{d} in the exponent. As in the case of T_{ox}, the variation of N_{d} does not show spatial correlation, and thus N_{d} can be handled using a similar method as used for T_{ox} in the framework.

The gate oxide tunneling current density, J_{tunnel}, can be represented by the following analytical model [39]:

$$J_{\mathrm{tunnel}} = \frac{4\pi m^* q}{h^3}(kT)^2 \left(1 + \frac{\gamma kT}{2\sqrt{E_{\mathrm{B}}}}\right) \mathrm{e}^{\frac{E_{F0,\mathrm{Si/SiO_2}}}{kT}} \mathrm{e}^{-\gamma\sqrt{E_{\mathrm{B}}}} \tag{4.22}$$

Here m^* is the transverse mass that equals $0.19m_0$ for electron tunneling and $0.55m_0$ for hole tunneling, where m_0 is the free electron rest mass; h is Planck's constant; γ is defined as $4\pi T_{\mathrm{ox}}\sqrt{2m_{\mathrm{ox}}}/h$, where m_{ox} is the effective electron (hole) mass in the oxide; E_{B} is the barrier height; $E_{F0,Si/SiO_2} = q\phi_S - q\phi_F - E_G/2$ is the Fermi level at the $\mathrm{Si/SiO_2}$ interface, where ϕ_{S} is surface potential, ϕ_{F} is the Fermi energy level potential, either in the Si substrate for the gate tunneling current through the channel, or in the source/drain region for the gate tunneling current through the source/drain overlap; and E_{G} is the Si band gap energy.

However, this formulation (Equation 4.22) does not lend itself easily to the analysis of the effects of parameter variations. Therefore, we again use an empirically characterized model to estimate I_{gate} per transistor width through curve-fitting:

$$I_{\mathrm{gate}} = c' \times \mathrm{e}^{b_1 + b_2 L_{\mathrm{eff}} + b_3 L_{\mathrm{eff}}^2 + b_4 T_{\mathrm{ox}} + b_5 T_{\mathrm{ox}}^2} \tag{4.23}$$

where c' and the b_i terms are the fitting coefficients.

As before, under the variations of L_{eff} and T_{ox}, $I_{\mathrm{gat}e}$ can be approximated by applying first-order Taylor expansion to the exponent U' of Equation (4.23):

$$I_{\mathrm{gate}} = c' \times \mathrm{e}^{U'{}_0 + \lambda_1 \cdot \Delta L_{\mathrm{eff}} + \lambda_2 \cdot \Delta T_{\mathrm{ox}}} \tag{4.24}$$

where $U'{}_0$ is the nominal value of the exponent U', and λ_0 and λ_1 are the derivatives of U' to L_{eff} and T_{ox} evaluated at their nominal values, respectively.

Under this approximation, I_{gate} is also a lognormally distributed random variable, and its PDF can be characterized through the values of c', U'_0, and $\lambda_i{}'$. Since the gate tunneling leakage current is input state dependent, the PDFs of the I_{gate} variables are characterized for all possible input states for each type of gate in the library, and a simple look-up table (LUT) is used for model retrieval while evaluating a specific circuit.

4.4.2.1 Distribution of the Full-Chip Leakage Current

We now present an approach for finding the distribution of $I_{\mathrm{tot}}^{\mathrm{avg}}$ as formulated in Equation (4.18), which is a weighted sum of the subthreshold and gate leakage values for each gate, over all input patterns to the gate. Since the probability of each $\mathrm{vec}_{i,k}$ can be computed by specifying signal probabilities at the circuit primary inputs and propagating the probabilities into all gate pins in the circuit using routine techniques, in this section, we focus on the computation of the PDF of the weighted sum.

As each of $I_{\text{sub},k}$ (vec$_{i,k}$) or $I_{\text{gate},k}$ (vec$_{i,k}$) has a lognormal distribution, it can easily be seen that any multiplication by a constant maintains this property. Therefore, the problem of calculating the distribution of $I_{\text{tot}}^{\text{avg}}$ becomes that of computing the PDF of the sum of a set of lognormal random variables. Furthermore, the set of lognormal random variables in the summation could be correlated since:

- the leakage current random variables for any two gates may be correlated due to spatial correlations of intra-die variations of process parameters.
- within the same gate, the subthreshold and gate tunneling leakage currents are correlated, and the leakage currents under different input vectors are correlated, because they are sensitive to the same process parameters of the same gate, regardless of whether these are spatially correlated or not.

Theoretically, the sum of several lognormal distributed random variables is not known to have a closed form. However, it may be well approximated as a lognormal, as is done in Wilkinson's method [40].[4] That is, the sum of m lognormals, $S = \sum_{i=1}^{m} \mathrm{e}^{Y_i}$, where each Y_i is a normal random variable with mean m_{y_i} and standard deviation σ_{y_i}, and the Y_i variables can be correlated or uncorrelated, can be approximated as a lognormal e^Z, where Z is normally distributed, with mean m_z and standard deviation σ_z. In Wilkinson's approach, the values of m_z and σ_z are obtained by matching the first two moments, u_1 and u_2, of e^Z and S as follows:

$$u_1 = E(\mathrm{e}^Z) = E(S) = \sum_{i=1}^{m} E(\mathrm{e}^{Y_i}) \tag{4.25}$$

$$u_2 = E(\mathrm{e}^{2Z}) = E(S^2) = \mathrm{Var}(S) + E^2(S) \tag{4.26}$$

$$= \sum_{i=1}^{m} \mathrm{Var}(\mathrm{e}^{Y_i}) + 2\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} \mathrm{cov}(\mathrm{e}^{Y_i}, \mathrm{e}^{Y_j}) + E^2(S)$$

$$= \sum_{i=1}^{m} \mathrm{Var}(\mathrm{e}^{Y_i}) + 2\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} \left(E(\mathrm{e}^{Y_i}\mathrm{e}^{Y_j}) - E(\mathrm{e}^{Y_i})E(\mathrm{e}^{Y_j})\right) + E^2(S)$$

where $E(.)$ and Var(.) are the symbols for the mean and variance values of a random variable, and cov(., .) represents the covariance between two random variables.

In general, the mean and variance of a lognormal random variable e^{X_i}, where X_i is normal distributed with mean m_{x_i} and standard deviation σ_{x_i}, is computed by:

[4] An approximation of the sum of correlated lognormal random variables by Monte Carlo simulations is computationally difficult for large-sized problems. As an alternative, three analytical approaches have been overviewed and compared in [40]: Wilkinson's approach, Schwartz and Yeh's approach, and the cumulant-matching approach. Through numerical comparisons, [40] concluded that Wilkinson's method is the best in terms of computational simplicity and accuracy.

$$E(\mathrm{e}^{X_i}) = \mathrm{e}^{m_{x_i}+\sigma_{x_i}^2/2} \tag{4.27}$$

$$\mathrm{Var}(\mathrm{e}^{X_i}) = \mathrm{e}^{2m_{x_i}+2\sigma_{x_i}^2} - \mathrm{e}^{2m_{x_i}+\sigma_{x_i}^2} \tag{4.28}$$

The covariance between two lognormal random variables e^{X_i} and e^{X_j} can be computed by:

$$\mathrm{cov}(\mathrm{e}^{X_i}, \mathrm{e}^{X_j}) = E(\mathrm{e}^{X_i} \cdot \mathrm{e}^{X_j}) - E(\mathrm{e}^{X_i})E(\mathrm{e}^{X_j}) \tag{4.29}$$

Superposing Equations (4.27), (4.28), and (4.29) into Equations (4.25) and (4.26) results in:

$$u_1 = E(\mathrm{e}^{Z}) = \mathrm{e}^{m_z+\sigma_z^2/2} = E(S) = \sum_{i=1}^{m} (\mathrm{e}^{m_{y_i}+\sigma_{y_i}^2/2}) \tag{4.30}$$

$$u_2 = E(\mathrm{e}^{2Z}) = \mathrm{e}^{2m_z+2\sigma_z^2} = E(S^2) \tag{4.31}$$

$$= \sum_{i=1}^{m} (\mathrm{e}^{2m_{y_i}+2\sigma_{y_i}^2} - \mathrm{e}^{2m_{y_i}+\sigma_{y_i}^2}) + 2\sum_{i=1}^{m-1}\sum_{j=i+1}^{m} \Big(\mathrm{e}^{m_{y_i}+m_{y_j}+(\sigma_{y_i}^2+\sigma_{y_j}^2+2r_{ij}\sigma_{y_i}\sigma_{y_j})/2}$$
$$-\mathrm{e}^{m_{y_i}+\sigma_{y_i}^2/2}\mathrm{e}^{m_{y_j}+\sigma_{y_j}^2/2}\Big) + u_1^2$$

Where r_{ij} is the correlation coefficient between Y_i and Y_j.

Solving (Equation 4.30) and (Equation 4.31) for m_z and σ_z yields:

$$m_z = 2\ln u_1 - \frac{1}{2}\ln u_2 \tag{4.32}$$

$$\sigma_z^2 = \ln u_2 - 2\ln u_1 \tag{4.33}$$

The computational complexity of Wilkinson's approximation can be analyzed through the cost of computing m_z and σ_z. The computational complexities of m_z and σ_z are determined by those of u_1 and u_2, whose values can be obtained using the formulas in (Equation 4.30) and (Equation 4.31). It is clear that the computational complexity of u_1 is dominated by that of u_2, since the complexity of calculating u_1 is $O(m)$, while that of u_2 is $O(m \cdot N_{\mathrm{corr}})$, where N_{corr} is the number of correlated pairs among all pairs of Y_i variables. The cost of computing u_2 can also be verified by examining the earlier expression of u_2 in (Equation 4.26), in which the second term in the summation, in fact, corresponds to the covariance of Y_i and Y_j, which becomes zero when Y_i and Y_j are uncorrelated. Therefore, if $r_{ij} \neq 0$ for all pairs of Y_i and Y_j, the complexity of calculating u_2 is $O(m^2)$; if $r_{ij} = 0$ for all pairs of i and j, the complexity is $O(m)$.

As explained earlier, for full-chip leakage analysis, the number of correlated lognormal distributed leakage components in the summation could be extremely large, which could lead to a prohibitive amount of computation. If Wilkinson's method is applied directly, when the total number of gates in the circuit is N_{g}, the complexity for computing the sum will be $O(N_{\mathrm{g}}^2)$, which is impractical for large circuits. In the remainder of this section, we will propose to compute the summation in a more efficient way.

4.4.2.2 Reducing the Cost of Wilkinson's Method

Since Wilkinson's method has a quadratic complexity with respect to the number of correlated lognormals to be summed, we now introduce mechanisms to reduce the number of correlated lognormals in the summation to improve the computational efficiency.

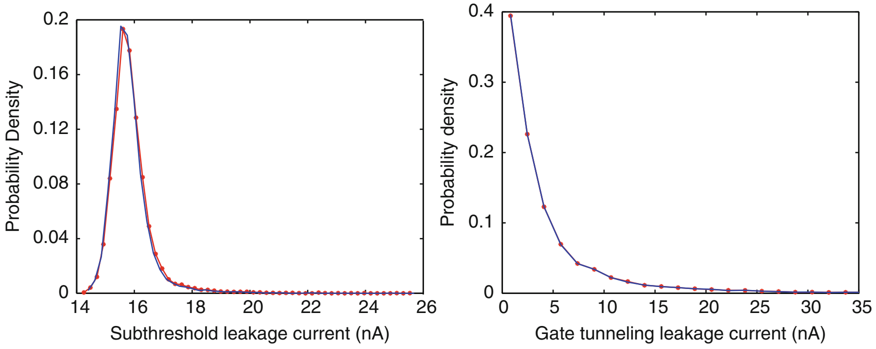
The work of [41] proposes a PCA-based method to compute the full-chip leakage considering the effect of spatial correlations of L_{eff}. The leakage current of each gate is rewritten in terms of its principal components by expanding the variable ΔL_{eff} as a linear function of principal components, i.e.,

$$I_{\mathrm{sub}}^i = \mathrm{e}^{U_{0,i} + \sum_{t=1}^{N_{\mathrm{p}}} \beta_{1,i} k_t^i \cdot p_t + \beta_{2,i} \cdot \Delta T_{\mathrm{ox},i}} \tag{4.34}$$

where N_{p} is the number of principal components. The sum of such lognormal terms can be approximated as a lognormal using Wilkinson's formula. The benefit of using a PCA form is that the mean and variance of a lognormal random variable can be computed in $O(N_{\mathrm{p}})$, as can the covariance of two lognormal random variables in PCA form. Therefore, the computation of all values and coefficients in I_{sub}^h, and thus the sum of two lognormals in PCA form, can be computed in $O(N_{\mathrm{p}})$. As mentioned in the description of Wilkinson's method, the computation of full-chip leakage current distribution requires a summation of N_{g} correlated lognormals. Thus, the PCA-based method has an overall computational complexity of $O(N_{\mathrm{p}} \cdot N_{\mathrm{g}})$.

A second approach, presented in [42], which we refer to as the "grouping method," uses two strategies for reducing the computations in applying Wilkinson's formula. First, the number of terms to be summed is reduced by identifying dominant states [38, 43] for the subthreshold and gate tunneling leakage currents for each type of gate in the circuit. As shown in Fig. 4.7a, the leakage PDF curves for simulations using dominant states only, and using the full set of states, for the average subthreshold leakage current of a three-input NAND gate are virtually identical. Similar results are seen for other gate types.

Second, instead of directly computing the sum of random variables of all leakage current terms, by grouping leakage current terms by model and grid location, and calculating the sum in each group separately first, the computational complexity in the computation of full-chip leakage reduces to quadratic in the number of groups. The key idea here is to characterize the leakage current per unit width for each stack type (called a model – these are N_{models} in number). The summation can be grouped by combining similar models in the same grid. Each group summation can be computed in linear time with respect to the number of leakage terms in the group. The results of the sums in all groups are then approximated as correlated lognormal



(a) Average subthreshold leakage current (b) Average gate-tunneling leakage current

Fig. 4.7 Comparison of PDFs of average leakage currents using dominant states with that of full input vector states for a 3-input NAND gate, by Monte Carlo simulation with 3σ variations of L_{eff} and T_{ox} 20%. The solid curve shows the result when only dominant states are used, and the starred curve corresponds to simulation with all input vector states

random variables that can then be computed directly using Wilkinson's method, so that we must perform the summation over $N_{\mathrm{groups}} = N_{\mathrm{models}} N_{\mathrm{g}}$ terms. Since the number of groups is relatively small, a calculation that is quadratic in the number of groups is practically very economical.

Specifically, the computational complexity for estimating the distribution of full-chip leakage current is reduced from $O(N_{\mathrm{g}}^2)$ for a naïve application of Wilkinson's formula to a substantially smaller number $O(N_{\mathrm{models}}^2 \cdot n^2)$, where n is the number of correlation grid squares.

A third approach [44], called the "hybrid method," combines the PCA-based and grouping methods, which attack the problem in orthogonal ways. As in the second approach, the leakage of each group is computed in terms of the original random variables. During the summation over all groups, the PCA approach is used to reduce the overall cost. The results in this paper show that the second approach outperforms the first, and that the third (hybrid) method outperforms the second as the number of grid squares, n, becomes larger.

The results of full-chip leakage estimation are presented in Fig. 4.8, which show the distribution of total circuit leakage current achieved using a statistical approach (the accuracy of the three methods is essentially indistinguishable) and using Monte Carlo simulation for circuit c7552: it is easy to see that the curve achieved by the basic method matches well with the Monte Carlo simulation result. For all test cases, the run-time of these methods is in seconds or less, while the Monte Carlo simulation takes considerably longer: for the largest test case, c7552, this simulation takes 3 h.

In terms of accuracy, the three methods are essentially very similar. However, they differ in terms of run-time efficiencies. In Tables 4.1 and 4.2, we show the run-times for different methods for ISCAS85 and ISCAS89 benchmark sets,

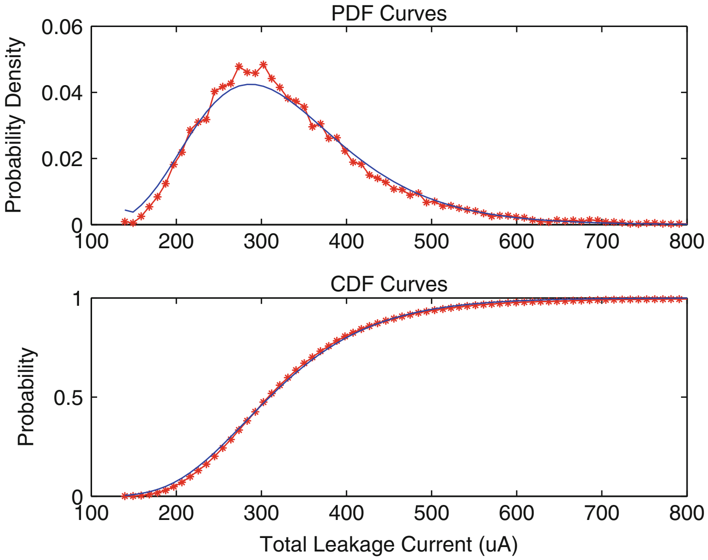


Fig. 4.8 Distributions of the total leakage against Monte Carlo simulation method for circuit c7552. The solid line illustrates the result of the proposed grouping method, while the starred line shows the Monte Carlo simulation results

Table 4.1 Run-time comparison of the PCA-based, grouping and hybrid methods for the ISCAS85 benchmarks

Benchmark	c432	c880	c1908	c2670	c3540	c6288	c5315	c7552
Number of grids	4	4	16	16	16	16	64	64
PCA-based method (s)	0.03	0.06	0.18	0.27	0.40	0.57	1.43	1.82
Grouping method (s)	0.01	0.02	0.04	0.06	0.09	0.10	0.24	0.29
Hybrid method (s)	0.01	0.03	0.06	0.09	0.12	0.14	0.19	0.25

Table 4.2 Run-time comparison of the proposed PCA-based, grouping, and hybrid methods for the ISCAS89 benchmarks

Benchmark	s5378	s9234	s13207	s15850	s35932	s38584
Number of grids	64	64	256	256	256	256
PCA-based method (s)	0.93	1.62	7.58	8.97	17.38	24.28
Grouping method (s)	0.22	0.32	5.89	5.91	4.97	10.04
Hybrid method (s)	0.16	0.30	0.47	0.56	1.03	1.34

respectively. In general, the grouping method is about 3–4 times faster than the PCA-based method. As expected, the hybrid approach does not show any run-time advantage over the grouping method for smaller grid sizes. However, run-time of both the grouping and the PCA-based methods grows much faster with the grid size than the hybrid method. In Tables 4.1 and 4.2, when the number of grids grows to greater than 64, the hybird approach is about 100 times faster than the other approaches. Therefore, the run-time can be significantly improved by hybridizing the PCA-based with the grouping approach.

Follow-up work in [45] presents alternative ideas for speeding up the summation of these lognormals, introducing the idea of a virtual-cell approximation, which sums the leakage currents by approximating them as the leakage of a single virtual cell.

4.5 Statistical Optimization

Process variations can significantly degrade the yield of a circuit, and optimization techniques can be used to improve the timing yield. An obvious way to increase the timing yield of the circuit is to pad the specifications to make the circuit robust to variations, i.e., to choose a delay specification of the circuit that is tighter than the required delay. This new specification must be appropriately selected to avoid large area or power overheads due to excessively conservative padding.

The idea of statistical optimization is presented in Fig. 4.9, in a space where two design parameters, p_1 and p_2, may be varied. The upper picture shows the constant value contours of the objective function, and the feasible region where all constraints are met. The optimal value for the deterministic optimization problem is the point at which the lowest value contour intersects the feasible set, as shown. However, if there is a variation about this point that affects the objective function, then after

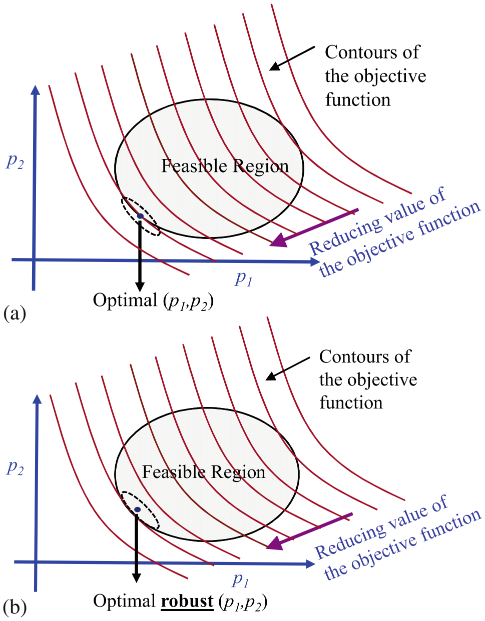

Contours of the objective function
p_2
Feasible Region
Reducing value of the objective function
p_1
Optimal (p_1,p_2)
(a)
Contours of the objective function
p_2
Feasible Region
Reducing value of the objective function
p_1
Optimal **robust** (p_1,p_2)
(b)

Fig. 4.9 A conceptual picture of robust optimization

manufacturing, the parameters may shift from the optimal design point. The figure shows an ellipsoidal variational region (corresponding to, say, the 99% probability contours of a Gaussian distribution) around an optimal design point: the manufactured solution may lie within this with a very high probability. It can be seen that a majority of points in this elliptical variational region lie outside the feasible set, implying a high likelihood that the manufactured circuit will fail the specifications. On the other hand, the robust optimum, shown in the lower picture, will ensure that the entire variational region will lie within the feasible set.

Therefore, statistical optimization is essentially the problem of determining the right amount by which the specifications should be "padded" in order to guarantee a certain yield, within the limitations of the process models. Too little padding can result in low yield, while too much padding can result in high resource overheads. More precisely, real designs are bounded from both ends. If the delays are too large, then the timing yield goes down, and if the delays are too small, this may be because of factors such as low threshold voltages in the manufactured part: in such a case, the leakage power becomes high enough that the part will fail its power specifications.

In the remainder of this section, we will first introduce techniques for finding statistical sensitivities – a key ingredient of any optimization method – and then overview some techniques for statistical optimization.

4.5.1 Statistical Sensitivity Calculation

A key problem in circuit optimization is the determination of statistical timing sensitivities and path criticality. Efficient computational engines for sensitivity analysis play an important role in guiding a range of statistical optimizations.

A straightforward approach in [46] involves perturbing gate delays to compute their effect on the circuit output delay. The complexity of the computation is reduced using the notion of a cutset belonging to a node in the timing graph: it is shown that the statistical maximum of the sum of arrival and required times across all the edges of a cutset gives the circuit delay distribution. If all sensitivities are to be computed, the complexity of this approach is potentially quadratic in the size of the timing graph.

For comprehensive sensitivity computation, one of the earliest attempts to compute edge criticalities was proposed in [30], which performs a reverse traversal of the timing graph, multiplying criticality probabilities of nodes with local criticalities of edges. However, this assumes that edge criticalities are independent, which is not a valid in practice. Follow-up work by the same group in [47] extends the cutset-based idea in [46] to compute the criticality of edges by linearly traversing the timing graph. The criticality of an edge in a cutset is computed using a balanced binary partition tree. Edges recurring in multiple cutsets are recorded in an array-based structure while traversing the timing graph.

Another effort in [48] approaches the problem by defining the statistical sensitivity matrix of edges in the timing graph with respect to the circuit output delay, and uses the chain rule to compute these values through a reverse traversal of the timing graph. Due to the matrix multiplications involved, albeit typically on sparse matrices, the complexity of the approach could be large, especially if the principal components are not sparse.

Like [46, 47], the work in [49] proposes an algorithm to compute the criticality probability of edges (nodes) in a timing graph using the notion of cutsets. Edges crossing multiple cutsets are dealt with using a zone-based approach, similar to [50], in which old computations are reused to the greatest possible extent. This work shows that without appropriate reordering, the errors propagated during criticality computations that use to Clark's *MAX* operation can be large; this is an effect that was ignored by previous approaches. Further, the work proposes a clustering-based pruning algorithm to control this error, eliminating a large number of non-competing edges in cutsets with several thousand edges. An extension in [51] investigates the effect of independent random variations on criticality computation and devises a simple scheme to keep track of structural correlations due to such variations.

4.5.2 Performance Optimization

Gate sizing is a valuable tool for improving the timing behavior of a circuit. In its most common form, it attempts to minimize an objective function, such as the area or the power dissipation, subject to timing constraints. In the literature, it is perhaps the most widely used target for statistical approaches, primarily because it is a transform that is applied at the right level, where design uncertainty does not overwhelm process uncertainty.

Early approaches to variation-tolerant gate sizing, which incorporate statistical timing models, include early work in [26], which formulates a statistical objective and timing constraints and solves the resulting nonlinear optimization formulation. However, this is computationally difficult and does not scale to large circuits. Other approaches for robust gate sizing that lie in the same family include [46, 52–54]: in these, the central idea is to capture the delay distributions by performing a statistical static timing analysis (SSTA), as opposed to the traditional STA, and then use either a general nonlinear programming technique or statistical sensitivity-based heuristic procedures to size the gates. In [55], the mean and variances of the node delays in the circuit graph are minimized in the selected paths, subject to constraints on delay and area penalty.

More formal optimization approaches have also been used. Approaches for optimizing the statistical power of the circuit, subject to timing yield constraints, can be presented as a convex formulation, as a second-order conic program [56]. For the binning model, a yield optimization problem is formulated [57], providing a binning yield loss function that has a linear penalty for delay of the circuit exceeding the target delay; the formulation is shown to be convex.

A gate sizing technique based on robust optimization theory has also been proposed [58, 59]: robust constraints are added to the original constraints set by modeling the intra-chip random process parameter variations as Gaussian variables, contained in a constant probability density uncertainty ellipsoid, centered at the nominal values.

Several techniques in the literature go beyond the gate sizing transform. For example, algorithms for statistically aware dual threshold voltage and sizing are presented in [60, 61]. Methods for optimal statistical pipeline design are present in [62], which explores the tradeoff between the logic depth of a pipeline and the yield, as well as gate sizing. The work argues that delay-unbalanced pipelines may provide better yields than delay-balanced pipelines.

4.6 Sensors for Post-Silicon Diagnosis

With the aid of SSTA tools, designers can optimize a circuit before it is fabricated, in the expectation that it will meet the delay and power requirements after being manufactured. In other words, SSTA is a presilicon analysis technique used to determine the range of performance (delay or power) variations over a large population of dies. A complementary role, after the chip is manufactured, is played by post-silicon diagnosis, which is typically directed toward determining the performance of an individual fabricated chip based on measurements on that specific chip. This procedure provides particular information that can be used to perform post-silicon optimizations to make a fabricated part meet its specifications. Because presilicon analysis has to be generally applicable to the entire population of manufactured chips, the statistical analysis that it provides shows a relatively large standard deviation for the delay. On the other hand, post-silicon procedures, which are tailored to individual chips, can be expected to provide more specific information. Since tester time is generally prohibitively expensive, it is necessary to derive the maximum possible information through the fewest post-silicon measurements.

In the past, the interaction between presilicon analysis and post-silicon measurements has been addressed in several ways. In [63], post-silicon measurements are used to learn a more accurate spatial correlation model, which is fed back to the analysis stage to refine the statistical timing analysis framework. In [64], a path-based methodology is used for correlating post-silicon test data to presilicon timing analysis. In [57], a statistical gate sizing approach is studied to optimize the binning yield. Post-silicon debug methods and their interaction with circuit design are discussed in [65].

In this section, we will discuss two approaches to diagnosing the impact of process variations on the timing behavior of a manufactured part. In each case, given the original circuit whose delay is to be estimated, the primary idea is to determine information from specific on-chip test structures to narrow the range of the performance distribution substantially. In the first case, we use a set of ring oscillators, and in the second, we synthesize a representative critical path whose behavior tracks the worst-case delay of the circuit. In each case, we show how the results of a limited measurement can be used to diagnose the performance of the manufactured part. The role of this step is seated between presilicon SSTA and post-silicon full chip testing. The approaches used here combine the results of presilicon SSTA for the circuit with the result of a small number of post-silicon measurements on an individual manufactured die to estimate the delay of that particular die.

An example use case scenario for this analysis in the realm of post-silicon tuning. Adaptive Body Bias (ABB) [66–68] is a post-silicon method that determines the appropriate level of body bias to be applied to a die to influence its performance characteristics. ABB is typically a coarse-grained optimization, both in terms of the granularity at which it can be applied (typically on a per-well basis) as well as in terms of the granularity of the voltage levels that may be applied (typically, the separation between ABB levels is 50–100 mV). Current ABB techniques use a replica of a critical path to predict the delay of the fabricated chip, and use this to feed a phase detector and a counter, whose output is then used to generate the requisite body bias value. Such an approach assumes that one critical path on a chip is an adequate reflection of on-chip variations. In general, there will be multiple potential critical paths even within a single combinational block, and there will be a large number of combinational blocks in a within-die region. Choosing a single critical path as representative of all of these variations is impractical and inaccurate. In contrast, an approach based on these test structures implicitly considers the effects of all paths in a circuit (without enumerating them, of course), and provides a PDF that concretely takes spatially correlated and uncorrelated parameters into account to narrow the variance of the sample, and has no preconceived notions, prior to fabrication, as to which path will be critical. The 3σ or 6σ point of this PDF may be used to determine the correct body bias value that compensates for process variations.

A notable approach [69, 70] addresses the related problem of critical path identification under multiple supply voltages. Since the critical paths may change as the supply voltage is altered, this method uses a voltage sensitivity-based procedure to identify a set of critical paths that can be tested to characterize the operating frequency of a circuit. An extension allows for sensitive paths to be dynamically configured as ring oscillators. While the method does not explicitly address process variations, the general scheme could be extended for the purpose. Overall, this method falls under the category of more time-intensive test-based approaches, as against the faster sensor-based approach described in the rest of this section, and plays a complementary role to the sensor-based method in post-silicon test.

4.6.1 Using Ring Oscillator Test Structures

In this approach, we gather information from a small set of test structures such as ring oscillators (ROs), distributed over the area of the chip, to capture the variations of spatially correlated parameters over the die. The physical sizes of the test structures are small enough that it is safe to assume that they can be incorporated into the circuit using reserved space that may be left for buffer insertion, decap insertion, etc. without significantly perturbing the layout.

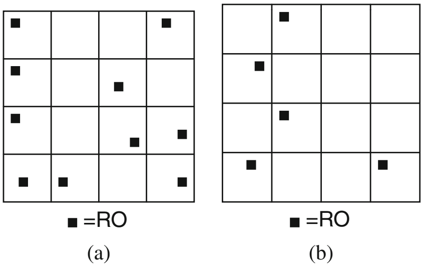


Fig. 4.10 Two different placements of test structures under the grid spatial correlation model

To illustrate the idea, we show a die in Fig. 4.10, whose area is gridded into spatial correlation regions. For simplicity, we will assume in this example that the spatial correlation regions for all parameters are the same, although the idea is valid, albeit with an uglier picture, if this is not the case. Fig. 4.10a,b show two cases where test structures are inserted on the die: the two differ only in the number and the locations of these test structures. The data gathered from the test structures in Fig. 4.10a,b are used in this paper to determine a new PDF for the delay of the original circuit, conditioned on this data. This PDF has a significantly smaller variance than that obtained from SSTA, as is illustrated in Fig. 4.11.

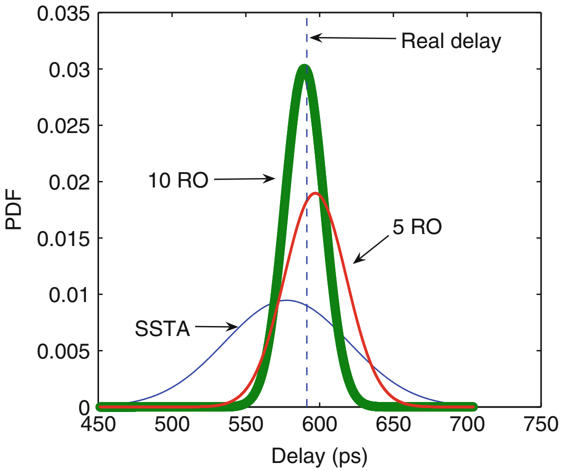


Fig. 4.11 Reduced-variance PDFs, obtained from statistical delay prediction, using data gathered from the test structures in Fig. 4.10

The plots in Fig. 4.11 may be interpreted as follows. When no test structures are used and no post-silicon measurements are performed, the PDF of the original circuit is the same as that computed by SSTA. When five ROs are used, a tighter spread is seen for the PDF, and the mean shifts toward the actual frequency for the die. This spread becomes tighter still when 10 ROs are used. In other words, as the number of test structures is increased, more information can be derived about variations on the die, and its delay PDF can be predicted with greater confidence: the standard deviation of the PDF from SSTA is always an upper bound on the standard deviation of this new delay PDF. In other words, by using more or fewer test structures, the approach is scalable in terms of statistical confidence.

If we represent the delay of the original circuit as d, then the objective is to find the conditional PDF of d, given the vector of delay values, $\mathbf{d}_r$, corresponding to the delays of the test structures, measured from the manufactured part. Note the $\mathbf{d}_r$ corresponds to one sample of the probabilistic delay vector, $\mathbf{d}_t$, of test structure delays. The corresponding means and variances of d are unsubscripted, and those of the test structures have the subscript "t."

We appeal to a well-known result to solve this problem: given a vector of jointly Gaussian distributions, we can determine the conditional distribution of one element of the vector, given the others. Specifically, consider a Gaussian-distributed vector $\begin{bmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{bmatrix}$ with mean $\boldsymbol{\mu}$ and a nonsingular covariance matrix $\boldsymbol{\Sigma}$. Let us define $\mathbf{X}_1 \sim N(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11}), \mathbf{X}_2 \sim N(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{22})$. If $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are partitioned as follows,

$$\boldsymbol{\mu} = \begin{bmatrix}\boldsymbol{\mu}_1\\ \boldsymbol{\mu}_2\end{bmatrix} \text{and} \quad \boldsymbol{\Sigma} = \begin{bmatrix}\boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{12}\\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22}\end{bmatrix}, \tag{4.35}$$

then the distribution of $\mathbf{X}_1$ conditional on $\mathbf{X}_2 = \mathbf{x}$ is multivariate normal, and its mean and covariance matrix are given by

$$\mathbf{X}_1|(\mathbf{X}_2 = \mathbf{x}) \sim N(\bar{\boldsymbol{\mu}}, \bar{\boldsymbol{\Sigma}}) \tag{4.36a}$$

$$\bar{\boldsymbol{\mu}} = \boldsymbol{\mu}_1 + \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}(\mathbf{x} - \boldsymbol{\mu}_2) \tag{4.36b}$$

$$\bar{\boldsymbol{\Sigma}} = \boldsymbol{\Sigma}_{11} - \boldsymbol{\Sigma}_{12}\boldsymbol{\Sigma}_{22}^{-1}\boldsymbol{\Sigma}_{21}. \tag{4.36c}$$

We define $\mathbf{X}_1$ as the *original subspace*, and $\mathbf{X}_2$ as the *test subspace*. By stacking d and $\mathbf{d}_t$ together, a new vector $\mathbf{d}_{\text{all}} = \left[d\ \mathbf{d}_t^{\text{T}}\right]^{\text{T}}$ is formed, with the original subspace containing only one variable d and the test subspace containing the vector $\mathbf{d}_t$. The random vector $\mathbf{d}_{\text{all}}$ is multivariate Gaussian-distributed, with its mean and covariance matrix given by:

$$\boldsymbol{\mu}_{\text{all}} = \begin{bmatrix}\mu\\ \boldsymbol{\mu}_t\end{bmatrix} \text{and} \quad \boldsymbol{\Sigma}_{\text{all}} = \begin{bmatrix}\sigma^2 & \mathbf{a}^{\text{T}}\mathbf{A}_t\\ \mathbf{A}_t^{\text{T}}\mathbf{a} & \boldsymbol{\Sigma}_t\end{bmatrix}. \tag{4.37}$$

We may then apply the above result to obtain the conditional PDF of d, given the delay information from the test structures. We know that the conditional distribution of d is Gaussian, and its mean and variance can be obtained as:

$$\text{PDF}(d_{\text{cond}}) = \text{PDF}\left(d|(\mathbf{d}_t = \mathbf{d}_r)\right) \sim N(\bar{\mu}, \bar{\sigma}^2) \tag{4.38a}$$

$$\bar{\mu} = \mu + \mathbf{a}^{\text{T}}\mathbf{A}_t\boldsymbol{\Sigma}_t^{-1}(\mathbf{d}_r - \boldsymbol{\mu}_t) \tag{4.38b}$$

$$\bar{\sigma}^2 = \sigma^2 - \mathbf{a}^{\mathrm{T}}\mathbf{A}_t\boldsymbol{\Sigma}_t^{-1}\mathbf{A}_t^{\mathrm{T}}\mathbf{a}. \tag{4.38c}$$

From Equations Equation (4.38b) and Equation (4.38c), we conclude that while the conditional mean of the original circuit is adjusted making use of the result vector, $\mathbf{d}_r$, the conditional variance is *independent* of the measured delay values, $\mathbf{d}_r$.

Examining Equation (4.38c) more closely, we see that for a given circuit, the variance of its delay before measuring the test structures, σ^2, and the coefficient vector, $\mathbf{a}$, are fixed and can be obtained from SSTA. The only variable that is affected by the test mechanism is the coefficient matrix of the test structures, $\mathbf{A}_t$, which also impacts $\boldsymbol{\Sigma}_t$. Therefore, the value of the conditional variance can be modified by adjusting the matrix $\mathbf{A}_t$. We know that $\mathbf{A}_t$ is the coefficient matrix formed by the sensitivities with respect to the principal components of the test structures. The size of $\mathbf{A}_t$ is determined by the number of test structures on the chip, and the entry values of $\mathbf{A}_t$ is related to the type of the test structures and their locations on the chip. Therefore if we use the same type of test structures on the circuit, then by varying their number and locations, we can modify the matrix $\mathbf{A}_t$, hence adjust the value of the conditional variance. Intuitively, this implies that the value of the conditional variance depends on how many test structures we have, and how well the test structures are distributed, in the sense of capturing spatial correlations between variables.

If the number of test structures equals the number of PCA components, the test structures collectively cover all principal components, and all variations are spatially correlated, then it is easy to show [71] that the test structures can exactly recover the principal components, and the delay of the manufactured part can be exactly predicted (within the limitations of statistical modeling). When we consider uncorrelated variations, by definition, it is impossible to predict these using any test structure that is disjoint from the circuit. However, we can drown these out by increasing the number of stages in the ring oscillator. This is shown in Fig. 4.12, which shows the effects of increasing the number of ring oscillator stages on predicting the delays of circuits s13207 and s5378. It is easily observed that the curves

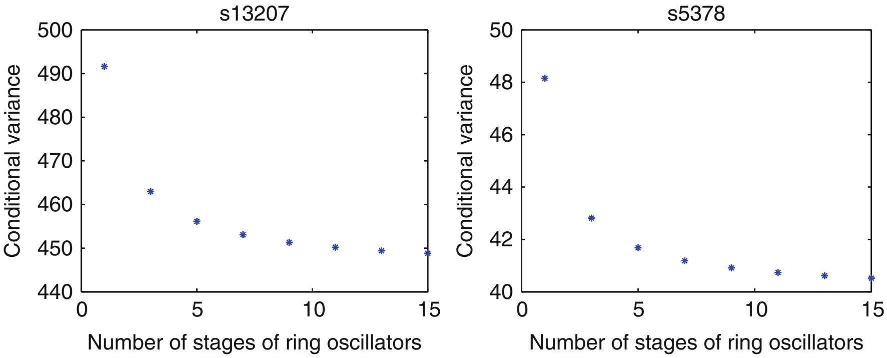


Fig. 4.12 Conditional variance of the delay of the original circuit with respect to the number of stages of ROs

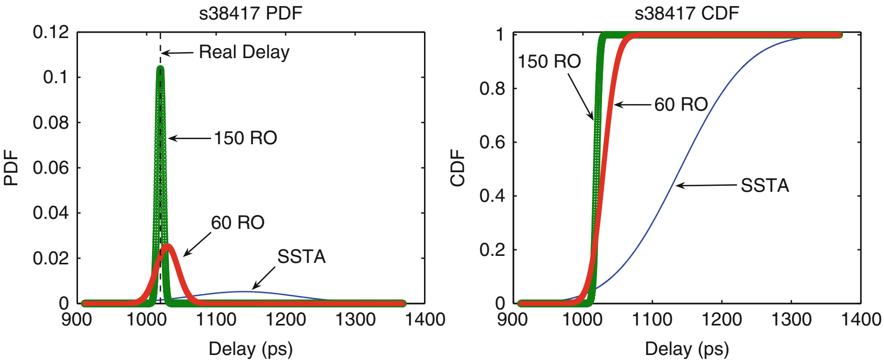


Fig. 4.13 PDF and CDF with insufficient number of test structures for circuit s38417 (considering L)

are monotonically decreasing. The results are similar for all other circuits in the benchmark set.

Finally, as was illustrated in Fig. 4.11, if a smaller number of test structures are used, then the variance of the conditional distribution increases. Figure 4.13 shows the predicted delay distribution for a typical sample of the circuit s38417, the largest circuit in the ISCAS89 benchmark suite. Each curve in the circuit corresponds to a different number of test structures, and it is clearly seen that even when the number of test structures is less than G, a sharp PDF of the original circuit delay can still be obtained using our method, with a variance much smaller than provided by SSTA. The tradeoff between the number of test structures and the reduction in the standard deviation can also be observed clearly. For this particular die, while SSTA can only assert that it can meet a 1400 ps delay requirement, using 150 test structures we can say with more than 99.7% confidence that the fabricated chip meets a 1040 ps delay requirement, and using 60 test structures we can say with such confidence that it can meet a 1080 ps delay requirement.

4.6.2 Using a Representative Critical Path

Another approach to post-silicon diagnosis involves the replication of a critical path of a circuit. As mentioned earlier, such techniques have been used in [66–68] in connection with adaptive body bias (ABB) or adaptive supply voltage (ASV) optimizations, where a replica of the critical path at nominal parameter values (we call this the *critical path replica* (CPR)) is used; its delay is measured to determine the optimal adaptation. However, such an approach has obvious problems: first, it is likely that a large circuit will have more than a single critical path, and second, a nominal critical path may have different sensitivities to the parameters than other near-critical paths, and thus may not be representative. An alternative approach in [71] uses a number of on-chip ring oscillators to capture the parameter variations of the original circuit. However, this approach requires measurements for hundreds of ring oscillators for a circuit with reasonable size and does not address issues related to how these should be placed or how the data can be interpreted online.

In this section, we describe how we may build an on-chip test structure that captures the effects of parameter variations on all critical paths, so that a measurement on this test structure provides us a reliable prediction of the actual delay of the circuit, with minimal error, for all manufactured die. The key idea is to synthesize the test structure such that its delay can reliably predict the maximum delay of the circuit, under across-die as well as within-die variations. In doing so, we take advantage of the property of spatial correlation between parameter variations to build this structure and determine the physical locations of its elements.

This structure, which we refer to as the *representative critical path* (RCP), is typically different from the critical path at nominal values of the process parameters. In particular, a measurement on the RCP provides the worst-case delay of the whole circuit, while the nominal critical path is only valid under no parameter variations, or very small variations. Since the RCP is an on-chip test structure, it can easily be used within existing post-silicon tuning schemes, e.g., by replacing the nominal critical path in the schemes in [66–68]. While our method accurately captures any correlated variations, it suffers from one limitation that is common to any on-chip test structure: it cannot capture the effects of spatially uncorrelated variations, because by definition, there is no relationship between those parameter variations of a test structure and those in the rest of the circuit. To the best of our knowledge, this work is the first effort that synthesizes a critical path in the statistical sense. The physical size of the RCP is small enough that it is safe to assume that it can be incorporated into the circuit (using reserved space that may be left for buffer insertion, decap insertion, etc.) without significantly perturbing the layout.

An obvious way to build an RCP is to use the nominal critical path for this prediction: this is essentially the critical path replica method [66–68]. However, the delay sensitivities of this nominal path may not be very representative. For instance, under a specific variation in the value of a process parameter, the nominal critical path delay may not be affected significantly, but the delay of a different path may be affected enough that it becomes critical. Therefore, we introduce the notion of building an RCP, and demonstrate that the use of this structure yields better results than the use of the nominal critical path.

The overall approach is summarized as follows. For the circuit under consideration, let the maximum delay be represented as a random variable, d_c. We build an RCP in such a way that its delay is closely related to that of the original circuit, and varies in a similar manner. The delay of this path can be symbolically represented by another random variable, d_p. Clearly, the ordered pair (d_c, d_p) takes on a distinct value in each manufactured part, and we refer to this value as ($d_\mathrm{cr}, d_\mathrm{pr}$). In other words, ($d_\mathrm{cr}, d_\mathrm{pr}$) corresponds to one sample of (d_c, d_p), corresponding to a particular set of parameter values in the manufactured part. Since the RCP is a single path, measuring d_{pr} involves considerably less overhead than measuring the delay of each potentially critical path. From the measured value of d_{pr}, we will infer the value, d_{cr}, of d_{c} for this sample, i.e., corresponding to this particular set of parameter values.

It can be shown mathematically [72] that in order to predict the circuit delay well, the correlation coefficient, ρ, between the RCP delay and the circuit delay must be high, i.e., close to 1. This is also in line with an intuitive understanding of the correlation coefficient. However, what is not entirely obvious is that this implies that the means of these delays can be very different, as long as ρ is high. In other words, we should try to match ρ rather than the mean delay, as is done when we choose the nominal critical path.

Assume that the circuit delay is listed in the canonical form in (Equation 4.13), and that the RCP delay d_{c} is also in canonical form as:

$$d_{\mathrm{c}} = \mu_c + \sum_{i=1}^{m} a_i p_i = \mu_c + \mathbf{a}^{\mathrm{T}}\mathbf{p} + R_c \tag{4.39}$$

where all terms inherit their meanings from Equation (4.13).

The correlation coefficient is then given by

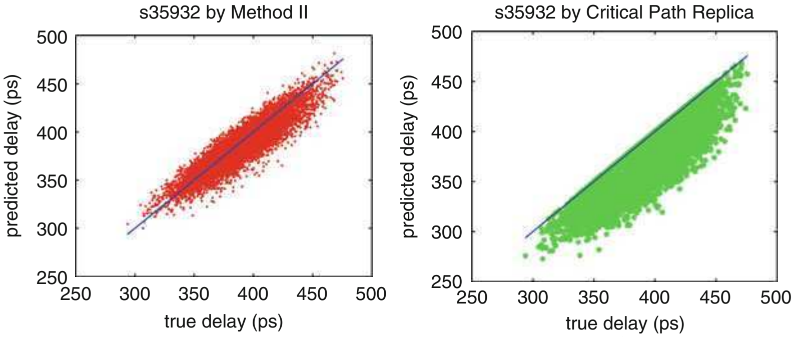
$$\rho = \frac{\mathbf{a}^{\mathrm{T}}\mathbf{b}}{\sigma_{\mathrm{c}}\sigma_{\mathrm{p}}} \tag{4.40}$$

where $\sigma_{\mathrm{c}} = \sqrt{\mathbf{a}^{\mathrm{T}}\mathbf{a} + \sigma_{R_c}^2}$ and $\sigma_{\mathrm{p}} = \sqrt{\mathbf{b}^{\mathrm{T}}\mathbf{b} + \sigma_{R_p}^2}$. An important point to note is that ρ depends only on the coefficients of the PCs for both the circuit and the critical path and their independent terms, and not on their means.

Although the problem of maximizing ρ can be formulated as a nonlinear programming problem, it admits no obvious easy solutions. Therefore, the work in [72] presents three heuristic approaches for finding the RCP. The first begins with the nominal critical path with all gates at minimum size, and then uses a greedy TILOS-like [73] heuristic to size up the transistors with the aim of maximizing ρ. The second builds the critical path from scratch, adding one stage at a time, starting from the output stage, each time greedily maximizing ρ as the new stage is added. The third combines these methods: it first builds the RCP using the second method, sets all transistors in it to minimum size, and then upsizes the transistors using a TILOS-like heuristic to maximize ρ greedily at each step.

The first method is cognizant of the structure of the circuit, and works well when the circuit is dominated by a single path, or by a few paths of similar sensitivity. When the number of critical paths is very large, choosing a single nominal path as a starting point could be misleading, and the second method may achieve greater benefits.

The results of the three methods are generally within similar ranges of accuracy. As expected, Method I performs better with circuits with a small number of critical paths, and Method II on circuits with more critical paths. Method III performs better



(a)

(b)

Fig. 4.14 The scatter plot: (**a**) true circuit delay vs. predicted delay by Method II and (**b**) true circuit delay vs. predicted delay using the CPR method

than Method II. With its more limited search space, Method II is the fastest of the three.

As an example result, we show scatter plots for both Method II and CPR for the circuit s35932 in Fig. 4.14a, b, respectively. The horizontal axis of both figures is the delay of the original circuit for a sample of the Monte Carlo simulation. The vertical axis of Fig. 4.14a is the delay predicted by our method, while the vertical axis of Fig. 4.14b is the delay of the nominal critical path, used by the CPR method. The ideal result is represented by the $(x = y)$ axis, shown using a solid line. It is easily seen that for the CPR method, the delay of the CPR is either equal to the true delay (when it is indeed the critical path of the manufactured circuit) or smaller (when another path becomes more critical, under manufacturing variations). On the other hand, for Method II, all points cluster closer to the $(x = y)$ line, an indicator that the method produces accurate results. The delay predicted by our approach can be larger or smaller than the circuit delay, but the errors are small. Note that neither Method II nor the CPR Method is guaranteed to be pessimistic, but such a consideration can be enforced by the addition of a guard band that corresponds to the largest error. The RCP approach has a clear advantage of a significantly smaller guard band in these experiments.

4.7 Conclusion

This chapter has presented an overview of issues related to the statistical analysis of digital circuits. Our focus has been on modeling statistical variations and carrying these into statistical timing and power analyses, which in turn are used to drive statistical optimization at the presilicon stage. Finally, we overview initial forays into the realm of using fast post-silicon measurements from special sensors to determine circuit delay characteristics.

Acknowledgments The author would like to thank Hongliang Chang, Qunzeng Liu, and Jaskirat Singh, whose work has contributed significantly to the content of this chapter.

References

1. Liu Y, Nassif SR, Pileggi LT, Strojwas AJ (Jun 2000) Impact of interconnect variations on the clock skew of a gigahertz microprocessor. In: Proceedings of the ACM/IEEE design automation conference, Los Angeles, CA, pp 168–171
2. Chang H, Sapatnekar SS (Nov 2003) Statistical timing analysis considering spatial correlations using a single PERT-like traversal. In: Proceedings of the IEEE/ACM international conference on computer-aided design, San Jose, CA, pp 621–625
3. Agarwal A, Blaauw D, Zolotov V, Sundareswaran S, Zhao M, Gala K, Panda R (Jan 2003) Statistical timing analysis considering spatial correlations. In: Proceedings of the Asia/South Pacific Design Automation Conference, Kitakyushu, pp 271–276
4. Agarwal A, Blaauw D, Zolotov V, Sundareswaran S, Zhao M, Gala K, Panda R (Dec 2002) Path-based statistical timing analysis considering inter- and intra-die correlations. In: Workshop notes, ACM/IEEE international workshop on timing issues in the specification and synthesis of digital systems (TAU), Monterey, CA, pp 16–21
5. Pelgrom MJM, Duinmaijer ACJ, Welbers APG (Oct 1989) Matching properties of MOS transistors. IEEE J Solid-State Circuits 24(5):1433–1440
6. Xiong J, Zolotov V, He L (Apr 2006) Robust extraction of spatial correlation. In: Proceedings of the ACM international symposium on physical design, San Jose, CA, pp 2–9
7. Liu F (Jun 2007) A general framework for spatial correlation modeling in VLSI design. In: Proceedings of the ACM/IEEE design automation conference, San Diego, CA, pp 817–822
8. Chen R, Zhang L, Zolotov V, Visweswariah C, Xiong J (Jan 2008) Static timing: back to our roots. In: Proceedings of the Asia/South Pacific design automation conference, Seoul, pp 310–315
9. Morrison DF (1976) Multivariate statistical methods. McGraw-Hill, New York, NY
10. Bhardwaj S, Vrudhula S, Ghanta P, Cao Y (Jul 2006) Modeling of intra-die process variations for accurate analysis and optimization of nano-scale circuits. In: Proceedings of the ACM/IEEE design automation conference, San Francisco, CA, pp 791–796
11. Bell T. An ICA page – papers, code, demos, links. Available at http://www.cnl.salk.edu/tony/ica.html (Accessed Sep 22, 2010)
12. Hyvärinen A, Oja E (2000) Independent component analysis: algorithms and applications. Neural Networks, 13(4–5):411–430
13. Singh J, Sapatnekar SS (2006) Statistical timing analysis with correlated non-gaussian parameters using independent component analysis. In: Proceedings of the ACM/IEEE design automation conference, Anaheim, CA, pp 155–160
14. Orshansky M, Keutzer K (Jun 2002) A general probabilistic framework for worst case timing analysis. In: Proceedings of the ACM/IEEE design automation conference, New Orleans, LA, pp 556–561
15. Deguchi Y, Ishiura N, Yajima S (Jun 1991) Probabilistic CTSS: analysis of timing error probability in asynchronous logic circuits. In: Proceedings of the ACM/IEEE design automation conference, San Francisco, CA, pp 650–655
16. Liou JJ, Cheng KT, Kundu S, Krstic A (Jun 2001) Fast statistical timing analysis by probabilistic event propagation. In: Proceedings of the ACM/IEEE design automation conference, Las Vegas, NV, pp 661–666
17. Devgan A, Kashyap CV (Nov 2003) Block-based statistical timing analysis with uncertainty. In: Proceedings of the IEEE/ACM international conference on computer-aided design, San Jose, CA, pp 607–614
18. Ramalingam A, Nam G-J, Singh AK, Orshansky M, Nassif SR, Pan DZ (Nov 2006) An accurate sparse matrix based framework for statistical static timing analysis. In: Proceedings of the IEEE/ACM international conference on computer-aided design, San Jose, CA, pp 231–236

19. Singhee A, Singhal S, Rutenbar RA (Nov 2008) Practical, fast Monte Carlo statistical static timing analysis: why and how. In: Proceedings of the IEEE/ACM international conference on computer-aided design, San Jose, CA, pp 190–195
20. Kanoria Y, Mitra S, Montanari A (Apr 2010) Statistical static timing analysis using Markov chain Monte Carlo. In: Proceedings of design, automation, and test in Europe, Dresden, pp 813–818
21. Chang H, Sapatnekar SS (Sep 2005) Statistical timing analysis under spatial correlations. IEEE Trans Comput Aided Des Integr Circ Syst 24(9):1467–1482
22. Chang H, Zolotov V, Narayan S, Visweswariah C (Jun 2005) Parameterized block-based statistical timing analysis with non-Gaussian parameters, nonlinear delay functions. In: Proceedings of the ACM/IEEE design automation conference, Anaheim, CA, pp 71–76
23. Zhan Y, Strojwas A, Li X, Pileggi L (Jun 2005) Correlation-aware statistical timing analysis with non-Gaussian delay distributions. In: Proceedings of the ACM/IEEE design automation conference, Anaheim, CA, pp 77–82
24. Feng Z, Li P, Zhan Y (Jun 2007) Fast second-order statistical static timing analysis using parameter dimension reduction. In: Proceedings of the ACM/IEEE design automation conference, San Diego, CA, pp 244–249
25. Berkelaar M (Dec 1997) Statistical delay calculation, a linear time method. In: Workshop notes, ACM/IEEE international workshop on timing issues in the specification and synthesis of digital systems, Austin, TX, pp 15–24
26. Jacobs E, Berkelaar MRCM (Mar 2000) Gate sizing using a statistical delay model. In: Proceedings of design, automation, and test in Europe, Paris, pp 283–290
27. Papoulis A (1991) Probability, random variables, and stochastic processes, 3rd edn. McGraw-Hill, New York, NY
28. Tsukiyama S, Tanaka M, Fukui M (Jan 2001) A statistical static timing analysis considering correlations between delays. In: Proceedings of the Asia/South Pacific design automation conference, Yokohama, pp 353–358
29. Clark CE (1961) The greatest of a finite set of random variables. Oper Res 9:85–91
30. Visweswariah C, Ravindran K, Kalafala K, Walker SG, Narayan S (Jun 2004) First-order incremental block-based statistical timing analysis. In: Proceedings of the ACM/IEEE design automation conference, San Diego, CA, pp 331–336
31. Jyu HF, Malik S, Devadas S, Keutzer KW (Jun 1993) Statistical timing analysis of combinational logic circuits. IEEE Trans VLSI Syst 1(2):126–137
32. Li X, Le J, Gopalakrishnan P, Pileggi LT (Nov 2004) Asymptotic probability extraction for non- normal distributions of circuit performance. In: Proceedings of the IEEE/ACM international conference on computer-aided design, San Jose, CA, pp 2–9
33. Acar E, Devgan A, Rao R, Liu Y, Su H, Nassif S, Burns J (Aug 2003) Leakage and leakage sensitivity computation for combinational circuits. In: Proceedings of the international symposium of low power electronic devices, Seoul, pp 96–99
34. Najm FN (Dec 1994) A survey of power estimation techniques in VLSI circuits. IEEE Trans VLSI Syst 2(4):446–455
35. Taur Y, Ning TH (1998) Fundamentals of modern VLSI devices. Cambridge University Press, New York, NY
36. Mukhopadhyay S, Roy K (Aug 2003) Modeling and estimation of total leakage current in nanoscaled CMOS devices considering the effect of parameter variation. In: Proceedings of the international symposium of low power electronic devices, Seoul, pp 172–175
37. Rao R, Srivastava A, Blaauw D, Sylvester D (Aug 2003) Statistical estimation of leakage current considering inter- and intra-die process variation. In: Proceedings of the international symposium of low power electronic devices, Seoul, pp 84–89
38. Sirichotiyakul S, Edwards T, Oh C, Zuo J, Dharchoudhury A, Panda R, Blaauw D (Jun 1999) Stand-by power minimization through simultaneous threshold voltage selection and circuit sizing. In: Proceedings of the ACM/IEEE design automation conference, New Orleans, LA, pp 436–441

39. Bowman KA, Wang L, Tang X, Meindl JD (Aug 2001) A circuit level perspective of the optimum gate oxide thickness. IEEE Trans Electron Devices 48(8):1800–1810
40. Abu-Dayya AA, Beaulieu NC (Jun 1994) Comparison of methods of computing correlated lognormal sum distributions and outages for digital wireless applications. In: IEEE 44th vehicular technology conference, vol 1, Stockholm, pp 175–179
41. Srivastava A, Shah S, Agarwal K, Sylvester D, Blaauw D, Director SW (Jun 2005) Accurate and efficient gate-level parametric yield estimation considering correlated variations in leakage power and performance. In: Proceedings of the ACM/IEEE design automation conference, Anaheim, CA, pp 535–540
42. Chang H, Sapatnekar SS (Jun 2005) Full-chip analysis of leakage power under process variations, including spatial correlations. In: Proceedings of the ACM/IEEE design automation conference, Anaheim, CA, pp 523–528
43. Lee D, Kwong W, Blaauw D, Sylvester D (Jun 2003) Analysis and minimization techniques for total leakage considering gate oxide leakage. In: Proceedings of the ACM/IEEE design automation conference, Anaheim, CA, pp 175–180
44. Chang H, Sapatnekar SS (Apr 2007) Prediction of leakage power under process uncertainties. ACM Trans Des Autom Electron Syst 12(2); Article 12:27 pp
45. Kim W, Do KT, Kim YH (Apr 2010) Statistical leakage estimation based on sequential addition of cell leakage currents. IEEE Trans VLSI Syst 18(4):602–615
46. Chopra K, Shah S, Srivastava A, Blaauw D, Sylvester D (Nov 2005) Parametric yield maximization using gate sizing based on efficient statistical power and delay gradient computation. In: Proceedings of the IEEE/ACM international conference on computer-aided design, San Jose, CA, pp 1023–1028
47. Xiong J, Zolotov V, Venkateswaran N, Visweswariah C (Jul 2006) Criticality computation in parameterized statistical timing. In: Proceedings of the ACM/IEEE design automation conference, San Francisco, CA, pp 63–68
48. Li X, Le J, Celik M, Pileggi LT (Nov 2005) Defining statistical sensitivity for timing optimization of logic circuits with large-scale process and environmental variations. In: Proceedings of the IEEE/ACM international conference on computer-aided design, San Jose, CA, pp 844–851
49. Mogal H, Qian H, Sapatnekar SS, Bazargan K (Nov 2007) Clustering based pruning for statistical criticality computation under process variations. In: Proceedings of the IEEE/ACM international conference on computer-aided design, San Jose, CA, pp 340–343
50. Yoshimura T, Kuh ES (Jan 1982) Efficient algorithms for channel routing. IEEE Trans Comput Aided Des Integr Circ Syst 1(1):25–35
51. Mogal H, Qian H, Sapatnekar SS, Bazargan K (Mar 2009) Fast and accurate statistical criticality computation under process variations. IEEE Trans Comput Aided Des Integr Circ Syst 28(3):350–363
52. Choi SH, Paul BC, Roy K (Jun 2004) Novel sizing algorithm for yield improvement under process variation in nanometer technology. In: Proceedings of the ACM/IEEE design automation conference, San Diego, CA, pp 454–459
53. Sinha D, Shenoy NV, Zhou H (Nov 2005) Statistical gate sizing for timing yield optimization. In: Proceedings of the IEEE/ACM international conference on computer-aided design, San Jose, CA, pp 1037–1042
54. Agarwal A, Chopra K, Blaauw D, Zolotov V (Jun 2005) Circuit optimization using statistical static timing analysis. In: Proceedings of the ACM/IEEE design automation conference, Anaheim, CA, pp 338–342
55. Raj S, Vrudhala SBK, Wang J (Jun 2004) A methodology to improve timing yield in the presence of process variations. In: Proceedings of the ACM/IEEE design automation conference, San Diego, CA, pp 448–453
56. Mani M, Devgan A, Orshansky M (Jun 2005) An efficient algorithm for statistical power under timing yield constraints. In: Proceedings of the ACM/IEEE design automation conference, Anaheim, CA, pp 309–314

57. Davoodi A, Srivastava A (Jul 2006) Variability driven gate sizing for binning yield optimization. In: Proceedings of the ACM/IEEE design automation conference, San Francisco, CA, pp 956–964
58. Singh J, Sapatnekar SS (Jan 2008) A scalable statistical static timing analyzer incorporating correlated non-Gaussian and Gaussian parameter variations. IEEE Trans Comput Aided Des Integr Circ Syst 27(1):160–173
59. Singh J, Nookala V, Luo T, Sapatnekar S (Jun 2005) Robust gate sizing by geometric programming. In: Proceedings of the ACM/IEEE design automation conference, Anaheim, CA, pp 315–320
60. Srivastava A, Sylvester D, Blaauw D (Jun 2004) Statistical optimization of leakage power consider process variations using dual-Vth and sizing. In: Proceedings of the ACM/IEEE design automation conference, San Diego, CA, pp 773–778
61. Srivastava A, Sylvester D, Blaauw D (Jun 2004) Power minimization using simultaneous gate sizing, dual Vdd and dual Vth assignment. In: Proceedings of the ACM/IEEE design automation conference, San Diego, CA, pp 783–787
62. Datta A, Bhunia S, Mukhopadhyay S, Banerjee N, Roy K (Mar 2005) Statistical modeling of pipeline delay and design of pipeline under process variation to enhance yield in sub-100 nm technologies. In: Proceedings of design, automation, and test in Europe, Munich, pp 926–931
63. Lee B, Wang L, Abadir MS (Jul 2006) Refined statistical static timing analysis through learning spatial delay correlations. In: Proceedings of the ACM/IEEE design automation conference, San Francisco, CA, pp 149–154
64. Wang L, Bastani P, Abadir MS (Jun 2007) Design-silicon timing correlation–a data mining perspective. In: Proceedings of the ACM/IEEE design automation conference, San Diego, CA, pp 385–389
65. Abranmovici M, Bradley P, Dwarakanath K, Levin P, Memmi G, Miller D (Jul 2006) A reconfigurable design-for-debug infrastructure for SoCs. In: Proceedings of the ACM/IEEE design automation conference, San Francisco, CA, pp 7–12
66. Tschanz JW, Kao JT, Narendra SG, Nair R, Antoniadis DA, Chandrakasan AP, De V (Nov 2002) Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage. IEEE J Solid-State Circuits 37:1396–1402
67. Tschanz JW, Narendra S, Nair R, De V (May 2003) Effectiveness of adaptive supply voltage and body bias for reducing the impact of parameter variations in low power and high performance microprocessors. IEEE J Solid-State Circuits 38:826–829
68. Tschanz JW, Narendra S, Keshavarzi A, De V (May 2005) Adaptive circuit techniques to minimize variation impacts on microprocessor performance and power. In: Proceedings of the IEEE international symposium on circuits and systems, Kobe, pp 23–26
69. Paul S, Krishnamurthy S, Mahmoodi H, Bhunia S (Nov 2007) Low-overhead design technique for calibration of maximum frequency at multiple operating points. In: Proceedings of the IEEE/ACM international conference on computer-aided design, San Jose, CA, pp 401–404
70. Paul S, Mahmoodi H, Bhunia S (Feb 2010) Low-overhead *Fmax* calibration at multiple operating points using delay-sensitivity-based path selection. ACM Trans Des Autom Electron Syst 15(2), Article 19:34 pp
71. Liu Q, Sapatnekar SS (Jun 2007) Confidence scalable post-silicon statistical delay prediction under process variations. In: Proceedings of the ACM/IEEE design automation conference, San Diego, CA, pp 497–502
72. Liu Q, Sapatnekar SS (Apr 2009) Synthesizing a representative critical path for post-silicon delay prediction. In: Proceedings of the International symposium on physical design, San Diego, CA, pp 183–190
73. Fishburn JP, Dunlop AE (Nov 1985) TILOS: a posynomial programming approach to transistor sizing. In: Proceedings of the IEEE international conference on computer-aided design, Santa Clara, CA, pp 326–328

Chapter 5
Low-Power and Variation-Tolerant Memory Design

Hamid Mahmoodi

Abstract This chapter discusses the impact of process variations on reliability of Static Random Access Memory (SRAM). An overview of SRAM and process variations in the context of SRAM is provided. The mechanisms of SRAM parametric failures are discussed. Models and methods of estimating parametric yield of SRAM array and failure probabilities of SRAM cell under process variations are presented. Design approaches for enhancing the yield of SRAM under process variations are discussed. The design approaches discussed include statistical joint optimization of SRAM cell sizing and redundancy, dynamic circuit techniques, post silicon adaptive repair techniques, and variation-tolerant SRAM peripherals. A self-repairing SRAM is discussed which adaptively adjusts its body bias to improve its reliability. Finally, a discussion on adaptive low-power and variation-tolerant SRAM design for multi-media applications is provided.

5.1 Introduction

Static Random Access Memory or SRAM is found in almost any electronic systems. SRAM is occupying an increasingly larger fraction of the total chip area in microprocessors. In SRAM, the memory cells are organized in a two-dimensional array as shown in Fig. 5.1a [35]. Each memory cell is a six-transistor cell as shown in Fig. 5.1b. Density is a major concern in memory design, and hence, the transistors in the SRAM cell are sized to be very small. The core of the SRAM cell is a cross-coupled pair of inverters ($P_\mathrm{R} - N_\mathrm{R}$ and $P_\mathrm{L} - N_\mathrm{L}$ in Fig. 5.1b). The remaining two transistors ($\mathrm{AX_L}$ and $\mathrm{AX_R}$ in Fig. 5.1b) are used for both read and write access to the cell. The gate of the access transistors of all the cells in a row are connected together forming a Word-Line (WL) and the bit-lines (BL and BR in Fig. 5.1b) of all the cells in a column are connected together forming differential bit-lines. A row

H. Mahmoodi (✉)
Department of Electrical and Computer Engineering, San Francisco State University,
San Francisco, CA, USA
e-mail: mahmoodi@sfsu.edu

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_5,

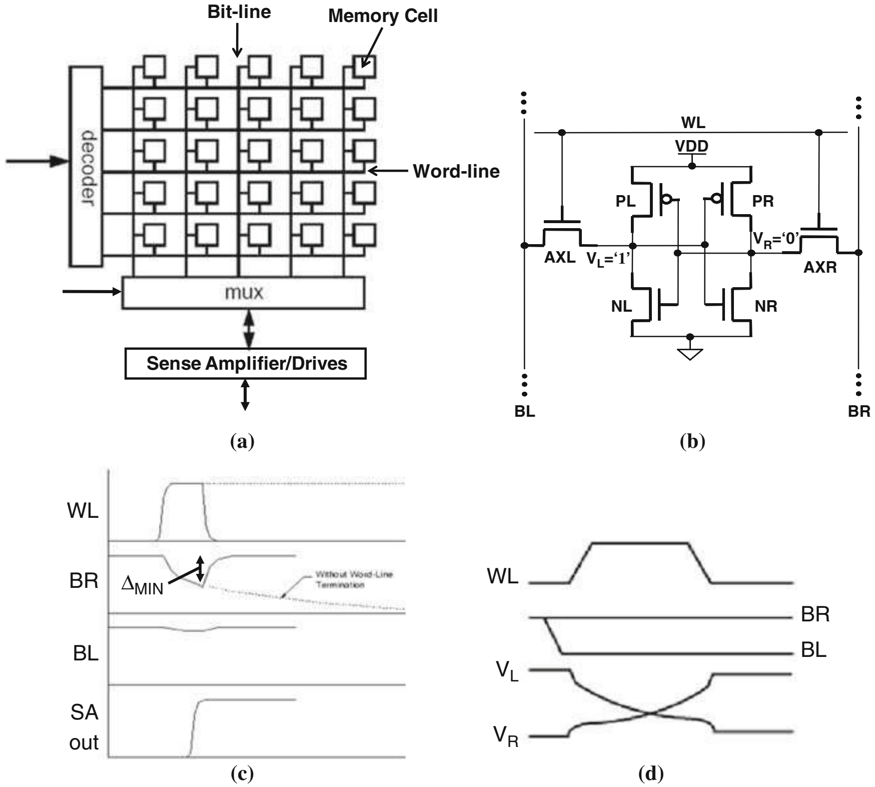


Fig. 5.1 (**a**) Organization of SRAM, (**b**) 6-transistor SRAM cell, (**c**) Read operation, (**d**) Write operation

decoder and a column multiplexer decode the address inputs, allowing selection of a unique cell for the read or write operation. The row decoder produces the WL signals and the column multiplexer selects the bit-line(s) to be accessed.

Figure 5.1c shows the timing of the read operation. For the read operation, first both the bit-lines are precharged to supply voltage (V_{DD}). Then, the decoder activates one word-line (WL switches to high). The cells in the selected word-line are connected to their bit-lines via the access transistors and start discharging their bit-lines differentially (one bit-line drops below V_{DD} and the other one remains at V_{DD} as shown in Fig. 5.1c). At the same time, the column multiplexer connects the selected bit-line(s) to sense amplifier(s) for full-swing amplification and output driving. Notice that there can be many cells connected to each bit-line in a column, and hence, the bit-line is a very capacitive line. Since the cell is not able to fully discharge the capacitive bit-line in the short time that the word-line is high, sense amplifiers are used to produce the full voltage swing outputs [35]. A critical design requirement is that the data of the cell should not flip after the read access. This can be achieved by proper transistor sizing as will be discussed later.

Figure 5.1d shows the waveforms of the write operation. For the write operation, the input data is differentially forced to the bit-lines (BL and BR) in full voltage swing and then a word-line is asserted. The cells in the activated row are updated with the forced data on the bit-lines. When the data put on the bit-lines is different from the initial data of the cell, the bit-line that is at low starts discharging the node of the cell which is at high and pulls it down to a voltage level sufficiently low for the cell to flip. Later in the chapter, the transistor sizing requirements for a successful write operation will be discussed.

Variations in process parameters have emerged as a major challenge in SRAM design. The sources of inter-die and intra-die variations in process parameters include variations in channel length, channel width, oxide thickness, threshold voltage, line edge roughness, and Random Dopant Fluctuations (RDF) [30] [2, 4, 41, 43]. RDF is the random variations in the number and location of dopant atoms in the channel region of the device resulting in the random variations in transistor threshold voltage [41]. Both the inter-die and intra-die process variations affect the stability and performance of the SRAM cell. The inter-die variation in a parameter (e.g., threshold voltage (V_t)) modifies the value of that parameter for all transistors of a die in the same direction. For example, threshold voltages of all the transistors of a die either increase or decrease from the nominal values as a result of inter-die variations. This principally results in a spread in the delay and the leakage parameters of dies, but does not cause mismatch between different transistors in a die. On the other hand, the intra-die variations shift the process parameters of different transistors in a die in different directions. For example, V_t of some transistors on a die increase whereas that of some others on the same die may decrease. The intra-die (or within-die) variations can be systematic or random. The systematic intra-die variations refer to correlated variations between neighboring transistors which happens when the shift in a parameter of one transistor depends on the shift of that parameter of a neighboring transistor. Random variations refer to uncorrelated variations which happen when shifts in a parameter of two neighboring transistors are completely independent. An example of the systematic intra-die variation can be the change in the channel length of different transistors of a die that are spatially correlated. The RDF induced V_t variation is a classic example of the random intra-die variation. The systematic variations do not result in large difference between the two transistors which are in close spatial proximity. The random component of the intra-die variation can result in significant mismatch between the neighboring transistors in a die [2, 4, 30, 41, 43].

In an SRAM cell, both inter and intra-die variations can result in the failure of the cell [5, 23, 24, 27]. SRAM cell failure can occur due to (a) an increase in the cell access time referred to as access time failure, (b) unstable read which is flipping of the cell data while reading, (c) unstable write which is inability to successfully write to a cell, or (d) failure in the data holding capability of the cell, which refers to flipping of the cell data at nominal supply voltage or with the application of a supply voltage lower than the nominal one in the standby mode. Since these failures are caused by the variations in the device parameters, these are known as the *parametric failures*. There can also be hard failures, caused by open or short, and soft failures due to soft error. This chapter concentrates only on the parametric failures, and hereafter the word “failure” refers to the parametric failures.

The remainder of this chapter is organized as follows: Section 5.2 briefly describes the different failure mechanisms in an SRAM cell. In Section 5.3, a modeling approach is discussed to estimate the parametric yield of an SRAM array in terms of the SRAM cell failure probability. In order to estimate the cell failure probability, three approaches are discussed: (a) standard Monte-Carlo simulation: this is the most accurate method but is the most complex; (b) sensitivity based modeling: it is simple and semi-analytical but is less accurate; and (c) important sampling Monte-Carlo: it is almost as accurate as standard Monte-Carlo method, but uses less number of samples by properly biasing the samples in failure regions. Section 5.4 discusses how to reduce cell failure probability and improve the yield. The discussion starts with statistical joint optimization of transistor sizing in SRAM cell and redundancy to enhance memory yield. Next, dynamic circuit techniques for improving yield are discussed. Finally, adaptive repair methods for yield enhancement are discussed. In particular, the architecture of a self-repairing SRAM array using adaptive body bias is discussed as an example of adaptive repair. Section 5.4 ends with a discussion on variation tolerance for SRAM peripherals. Section 5.5 discusses low power and variation tolerance in image and video application to trade-off between the image quality and the power. Finally the chapter is summarized and concluded in Section 5.6.

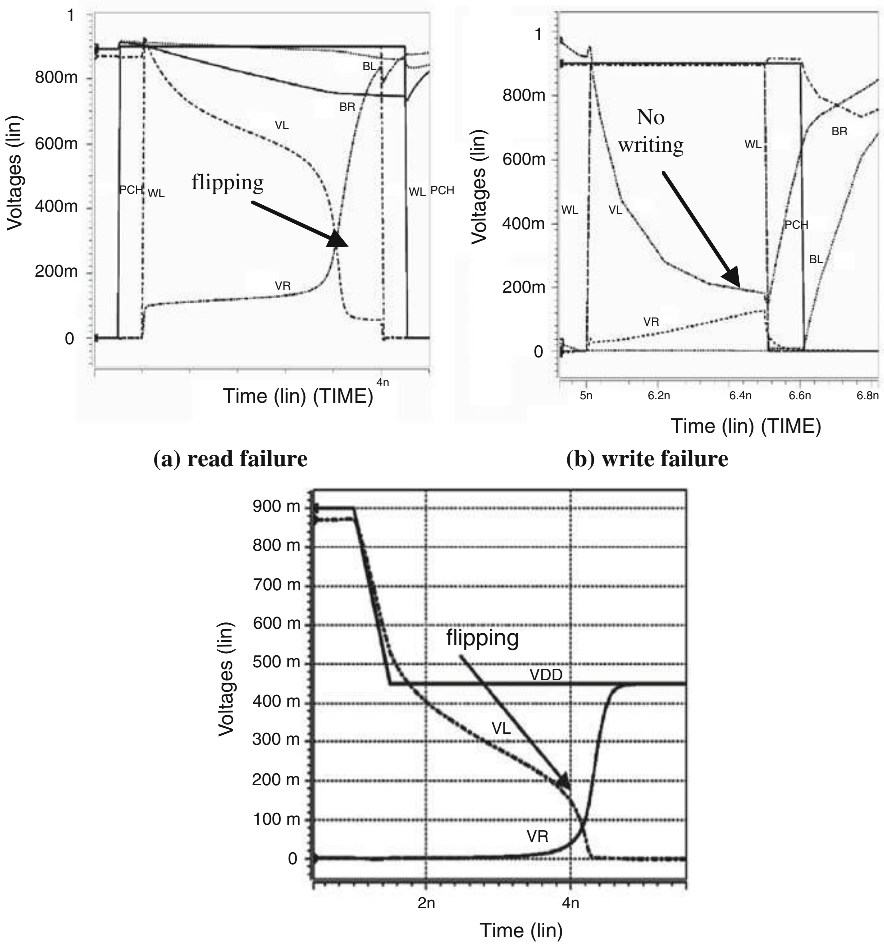
5.2 Mechanisms of Failure in SRAM Cell

Inter and intra-die variations in the process parameters (e.g., threshold voltage, channel length, channel width, etc. of transistors) result in uniform change or the mismatch in the strength of the different transistors in an SRAM cell (Fig. 5.1b). The uniform change as well as the device mismatch can result in the failure of the SRAM cell [24]. In this section, the mechanism of each of these failures in an SRAM cell is briefly described. The parametric failures in an SRAM cell are principally due to:

(1) Destructive read known as *read failure* (or read disturb failure). It refers to the flipping of the stored data in a cell while reading.
(2) Unsuccessful write or *write failure*. It refers to the inability to write to a cell.
(3) An increase in the access time of the cell resulting in a violation of delay requirement, defined as the *access time failure*.
(4) The destruction of the cell content in the standby mode at nominal supply voltage or with the application of a lower supply voltage, known as the *hold-stability failure*. A lower supply voltage may be applied in the standby mode to reduce leakage power.

5.2.1 Read Failure

Consider the cell in Fig. 5.1b (V_L=“1” and V_R=“0”) in the read mode when the word-line is at V_{DD} and the bit-lines are precharged to V_{DD}. While reading the cell, due to the voltage divider action between $\mathrm{AX_R}$ and N_R , the voltage at node $R(V_R)$ increases to a positive value called V_{READ}. If V_{READ} is higher than the trip point of the inverter $P_L - N_L(V_{\mathrm{TRIPRD}})$, then the cell flips while reading the cell as shown in Fig. 5.2a [6, 24]. This represents a read failure event. If the strength of the access transistor (AX_R) is higher than that of the pull-down NMOS transistor (N_R), the voltage division action between the two transistors increases the voltage V_{READ}. For example, due to the random variation in the threshold voltage, a reduction in the V_t (increase in strength) of the access transistor and an increase in the V_t (reduction in strength) of the pull-down NMOS results in an increase in V_{READ} from its nominal



(a) read failure

(b) write failure

(c) hold failure

Fig. 5.2 Unstable read, write, and hold failures in SRAM cell [24]. (**a**) read failure (**b**) write failure (**c**) hold failure

value, thereby resulting in a read failure. Similarly, trip-point of the inverter $P_{\mathrm{L}} - N_{\mathrm{L}}$ depends on the strengths of the pull-up PMOS and pull-down NMOS. Normally SRAM cell is designed to have a weaker PMOS, which results in a lower value of V_{TRIP}. Although the nominal value of V_{TRIP} is not less than the nominal value of V_{READ}, parameter variation can result in an increase in the V_{t} of P_{L} and/or reduction in the V_{t} of N_{L}. This can lower V_{TRIP} below V_{READ} thereby resulting in a read failure. It should be noted that the read failure is caused by the mismatch in the strength of the different transistors. This mismatch can only be caused by the effect of *random intra-die variation*. Hence, an increase in the random intra-die variation can significantly increase the read failure probability.

5.2.2 Write Failure

Consider the cell in Fig. 5.1b, storing logic state “1” (V_{L}=“1” and V_{R}= “0”). While writing a “0” to the cell, the node V_{L} gets discharged through BL to a low value (V_{WR}), which is determined by the voltage division between the PMOS P_{L} and the access transistor AX_{L} [6, 24]. If V_{L} cannot be reduced below the trip point of the inverter $P_{\mathrm{R}} - N_{\mathrm{R}}(V_{\mathrm{TRIPWR}})$, within the time when word-line is high (T_{WL}), then a write failure occurs as shown in Fig. 5.2b. The discharging current (I_{L}) at node L is the difference in the ON currents of the access transistor $AX_L(I_{AX_L})$ and the PMOS P_{L} (I_{PL}) (i.e., $I_{\mathrm{L}} = I_{\mathrm{AX_L}} - I_{\mathrm{PL}}$). Hence, a stronger PMOS and a weaker access transistor can significantly slow down the discharging process thereby causing a write failure. The variation in the device strengths due to inter-die as well as random variation in process parameters can increase the write time. For example, if V_{t} of P_{L} reduces and that of AX_{L} increases, that can result in an increase in the write time, thereby causing write failure.

5.2.3 Access Time Failure

Consider the cell in Fig. 5.1b, storing logic state “1” (V_{L}= “1” and V_{R}= “0”). Before the read operation begins the bit-lines (BR and BL) are precharged to V_{DD}. When the word-line is switched to high, the BR is discharge through AX_{R} and N_{R}. As shown in Fig. 5.1c, the cell is expected to develop sufficient voltage drop on BR ($\Delta_{\mathrm{MIN}} \approx 0.1\ V_{\mathrm{DD}}$) so that the sense amplifier can reliably detect the state of the cell and produce full swing voltage differential at its outputs [35]. The cell access time (T_{ACCESS}) is defined as the time required for producing the pre-specified voltage difference, Δ_{MIN}, between the two bit-lines. If due to V_{t} variation, the access time of the cell is longer than the maximum tolerable limit (T_{MAX}), an access time failure is said to have occurred. A consequence of access time failure is that a wrong value may be read and sent out by the sense amplifier during the read operation. Access time failure is caused by the reduction in the strength of the access and the pull-down transistors. An increase in the V_{t} of the access transistor and/or the pull-down

NMOS can significantly increase the access time from its nominal value creating access time failure. It should be noted that the access failure is caused by increase in the V_t of AX_R and/or V_t of N_R. Thus, both intra-die and inter-die variations in process parameters increase the access time failure.

5.2.4 Hold Stability Failure

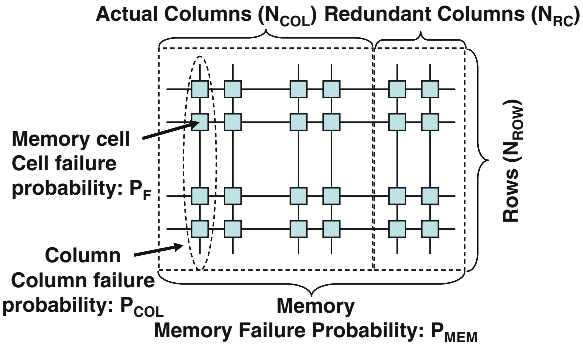
In the standby mode, the V_{DD} of the cell may be reduced to reduce the leakage power consumption [14]. However, if lowering of V_{DD} causes the data stored in the cell to be destroyed, then the cell is said to have failed in the hold mode [34]. As the supply voltage of the cell is lowered, the voltage at the node storing "1" (node L in Fig. 5.1b) also reduces. Moreover, for a low supply voltage and when P_L is not strongly "ON" due to process variation, leakage of the pull-down NMOS N_L reduces the voltage at node L below the supply voltage applied to the cell. If the voltage at the node L is reduced below the trip-point of the inverter $P_R - N_R$, then flipping occurs and the data is lost in the hold mode, as shown in Fig. 5.2c [24]. The supply voltage to be applied in the hold mode is chosen to ensure the holding of the data under nominal process condition. However, variation in the process parameter can result in the device mismatch causing hold failures. For example increase in V_t of P_L facilitates the reduction of the voltage at node L below the applied supply voltage. If the V_t of N_L reduces while that of P_L increases and/or V_t of N_R increases while that of P_R reduces (increase in the trip-point of $P_R - N_R$), the possibility of data flipping in the hold mode increases. Consequently, an increase in the random intra-die variation can significantly increase the hold failure probability.

5.3 SRAM Yield Estimation Methods

The hierarchy of the failure probabilities in an SRAM array is shown in Fig. 5.3. First, the failure probability of a cell (P_F) needs to be estimated. The cell failure probability (P_F) is then used to determine the probability of failure of a column (P_{COL}) which depends on the total number of cells in that column (i.e., column length). The estimated value of P_{COL} is then used to estimate the failure probability of the memory array (P_{MEM}). The failure probability of the memory array depends on the number of actual columns (N_{COL}) and the redundant columns (N_{RC}). The memory failure probability is directly related to the yield of the memory chip.

The yield is defined as the percentage of the passing chips. To estimate the yield, Monte-Carlo simulations need to be performed for *inter-die* distributions of L, W, and V_t. For each set of inter-die values of the parameters (say L_{INTER}, W_{INTER}, and $V_{t_{INTER}}$), the memory failure probability (P_{MEM}) need to be estimated considering the intra-die distribution of threshold voltage variation (δV_t). Finally, the yield is defined as:

Fig. 5.3 Memory hierarchy and failure probability [24]


Actual Columns (N_{COL}) Redundant Columns (N_{RC})
Memory cell
Cell failure probability: P_F
Column
Column failure probability: P_{COL}
Memory
Memory Failure Probability: P_{MEM}
Rows (N_{ROW})

$$\mathrm{Yield} = 1 - \left(\sum_{\mathrm{INTER}} P_{\mathrm{MEM}} \left(L_{\mathrm{INTER}}, W_{\mathrm{INTER}}, V_{\mathrm{t_{INTER}}} \right) \Big/ N_{\mathrm{INTER}} \right) \tag{5.1}$$

where N_{INTER} is the total number of inter-die Monte-Carlo simulations (i.e., total number of chips) [24].

Assuming column redundancy, the probability of failure of a memory chip (P_{MEM}) designed with N_{COL} number of columns and N_{RC} number of redundant columns is defined as the probability that more than N_{RC} (i.e., at least $N_{\mathrm{RC}}+1$) columns fail. Hence, P_{MEM} can be expressed in terms of column failure probability (P_{COL}) as follows:

$$P_{\mathrm{MEM}} = \sum_{i=N_{\mathrm{RC}}+1}^{N_{\mathrm{COL}}} \binom{N_{\mathrm{COL}}}{i} P_{\mathrm{COL}}^{i} \left(1 - P_{\mathrm{COL}}\right)^{N_{\mathrm{COL}}-i} \tag{5.2}$$

The failure probability of column (P_{COL}) (or row (P_{ROW})) is defined as the probability that any of the cells (out of N cells) in that column (or row) fails. Hence, P_{COL} can be expressed in terms of SRAM cell failure probability (P_{F}) by

$$P_{\mathrm{COL}} = 1 - (1 - P_{\mathrm{F}})^{N} \tag{5.3}$$

SRAM cell is considered to be faulty if any of the failure mechanisms (Access Time failure (AF), Read Failure (RF), Write Failure (WF), or Hold Failure (HF)) occur. Hence, the overall failure probability can be expressed by

$$\begin{aligned} P_{\mathrm{F}} = P[\mathrm{Fail}] = P[A_{\mathrm{F}} + R_{\mathrm{F}} + W_{\mathrm{F}} + H_{\mathrm{F}}] = P_{\mathrm{AF}} + P_{\mathrm{RF}} + P_{\mathrm{WF}} + P_{\mathrm{HF}} \\ - P[A_{\mathrm{F}}R_{\mathrm{F}}] - P[A_{\mathrm{F}}W_{\mathrm{F}}] - P[A_{\mathrm{F}}H_{\mathrm{F}}] - P[R_{\mathrm{F}}W_{\mathrm{F}}] - P[R_{\mathrm{F}}H_{\mathrm{F}}] - P[W_{\mathrm{F}}H_{\mathrm{F}}] \\ + P[A_{\mathrm{F}}R_{\mathrm{F}}W_{\mathrm{F}}] + P[A_{\mathrm{F}}R_{\mathrm{F}}H_{\mathrm{F}}] + P[R_{\mathrm{F}}W_{\mathrm{F}}H_{\mathrm{F}}] + P[W_{\mathrm{F}}H_{\mathrm{F}}A_{\mathrm{F}}] - P[\mathrm{All}] \end{aligned} \tag{5.4}$$

where P_{AF}, P_{RF}, P_{WF}, and P_{HF} represent the probability of access time failure, read failure, write failure, and hold failure events, respectively. An accurate estimate of the probability of joint events is possible by constructing the joint Probability

Density Function (PDF) representing two events [24]. It can also be assumed that probabilities of simultaneous occurrence of more than two events are negligible. The key to accurate estimation of SRAM yield is accurate estimation of probabilities of individual failure events under intra-die variations. In the remaining of this section, three approaches for estimation of failure probabilities are discussed.

5.3.1 Failure Probability Estimation Using Monte-Carlo Simulations

In a SRAM die, failures are principally caused by the mismatch caused by intra-die variations in the device parameters (L, W, V_{t}) of different transistors in the cell. Such device mismatch changes the strength of the different transistors resulting in different failure events. The principal source of the device mismatch is the intrinsic fluctuation of the V_{t} of different transistors due to Random Dopant Fluctuations (RDF) [2]. Hence, in this chapter, V_{t} variation due to RDF is considered as the major source of intra-die variation. The discussed method can also be extended to include L and W variation.

In an SRAM cell, the threshold voltage fluctuations (δV_{t}) of the six transistors are considered as six *independent Gaussian random variables* with mean values of 0 [2, 24]. The standard deviation of the V_{t} fluctuation (σV_{t}) depends on the manufacturing process, doping profile, and the transistor sizing. The dependence of (σV_{t}) on the transistor size (W and L) is given by [42]:

$$\sigma_{\delta Vt_i} = \left(qT_{\mathrm{ox}} \big/ \varepsilon_{\mathrm{ox}}\right) \sqrt{N_{\mathrm{SUB}} W_{\mathrm{dm}} \big/ 3LW} \tag{5.5}$$

where T_{ox} is the oxide thickness, W_{dm} is the width of the depletion region, and N_{SUB} is the doping concentration in substrate.

In the Monte-Carlo simulation method, a random assignment of δV_{t} is chosen for each transistor in the SRAM cell and the design is simulated (e.g., Spice simulation) to determine if there are any failures in the cell. After iterating this process for many times and counting the number of occurrences of the cell failure, the cell failure probability is estimated at the number of occurrences of cell failure divided by the total number of Monte-Carlo simulations. Given the small value of cell failure probability, this method requires many simulations to offer an acceptable accuracy, and hence, it is very time consuming [19]. This makes it impractical to use in real design environments where a designer has to try many design (e.g. transistor sizing) options for the cell in order to maximize the memory yield. Two other failure probability estimation methods are discussed in the remaining of this section that can help to reduce the estimation time. The first approach is a sensitivity-based approach [24] and the second method is importance sampling based rare-event Monte-Carlo approach [19].

5.3.2 *Failure Probability Estimation Using Sensitivity-Based Method*

In this section, first the key mathematical bases of the sensitivity-based method, used to estimate the failure probabilities, are summarized. Let us consider $y = f(x_1, \ldots, x_n)$ as a function, where $x_1, \ldots, x_n$ are *independent Gaussian random variables* with mean $\eta_1, \ldots, \eta_2$ and standard deviation (STD) $\sigma_1, \ldots, \sigma_2$. The mean ($\mu_y$) and the STD ($\sigma_y$) of the random variable y can be estimated as (using multi-variable Taylor-series expansion) [32]:

$$\begin{aligned}\mu_y &= f(\eta_1, ..., \eta_n) + \frac{1}{2}\sum_{i=1}^{n} \left.\frac{\partial^2 f(x_1,...,x_n)}{\partial(x_i)^2}\right|_{\eta_i} \sigma_i^2 \\ \sigma_y^2 &= \sum_{i=1}^{n} \left(\left.\frac{\partial f(x_1,...,x_n)}{\partial(x_i)}\right|_{\eta_i} \right)^2 \sigma_i^2\end{aligned} \tag{5.6}$$

Assuming the Probability Distribution Function (PDF) of y to be also Gaussian ($N_y(y : \mu_y,\ \sigma_y)$), the probability of ($y > Y_0$) is given by

$$P\left[y > Y_0\right] = \int_{y=Y_0}^{\infty} N_y(y : \mu_y, \sigma_y)dy = 1 - \int_{y=-\infty}^{Y_0} N_y(y)dy = 1 - \Phi_y\left(Y_0\right) \tag{5.7}$$

where Φ_y is the Cumulative Distribution Function (CDF) of y. The above mathematical basis (sensitivity-based method) is used in this section to estimate the failure probabilities of different events.

5.3.2.1 Read Stability Failure (RF)

As explained in Section 5.3.2, *read failure* occurs, when during reading, the voltage of the node R (V_{READ}) increases to a value higher than the trip-point of the inverter $P_{\mathrm{L}} - N_{\mathrm{L}}(V_{\mathrm{TRIPRD}})$ (Fig 5.2a). Hence, the read-failure probability (P_{RF}) is given by

$$P_{\mathrm{RF}} = P\left[V_{\mathrm{READ}} > V_{\mathrm{TRIPRD}}\right] \tag{5.8}$$

V_{TRIPRD} and V_{READ} can be obtained by solving Kirchhoff's Current Law (KCL) at nodes L and R, respectively. Assuming the PDF of $V_{\mathrm{READ}}(= N_{\mathrm{RD}}(v_{\mathrm{READ}}))$ and $V_{\mathrm{TRIP}}(= N_{\mathrm{TRIP}}(v_{\mathrm{TRIP}}))$ to follow Gaussian distributions with the means and the variances obtained using (Equation (5.6)) as shown in Fig. 5.4b, P_{RF} is given by

$$\begin{aligned}&P_{\mathrm{RF}} = P\left[Z_{\mathrm{R}} \equiv (V_{\mathrm{READ}} - V_{\mathrm{TRIPRD}}) > 0\right] = 1 - \Phi_{\mathrm{ZR}}(0) \\ &\text{where, } \eta_{ZR} = \eta_{V_{\mathrm{READ}}} - \eta_{V_{\mathrm{TRIP}}} \text{ and } \sigma_{\mathrm{ZR}}^2 = \sigma_{V_{\mathrm{READ}}}^2 + \sigma_{V_{\mathrm{TRIPRD}}}^2\end{aligned} \tag{5.9}$$

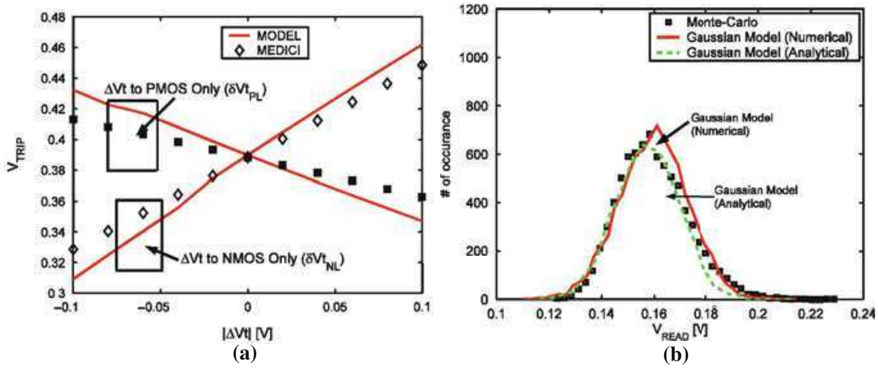


Fig. 5.4 (**a**) Variation of V_{TRIP} of $P_{\mathrm{L}} - N_{\mathrm{L}}$ (**b**) Distributions of V_{READ} [24]

In [24], authors have shown that variation of V_{TRIPRD} and V_{READ} estimated using the above approach can closely match with device simulations as shown in Fig. 5.4.

5.3.2.2 Write Stability Failure (WF)

Following the discussion in Section 5.3.2, the write failure occurs when, while writing a “0” to the node storing “1” (node L in Fig. 5.1b), the voltage at node L (V_{L}) is not reduced below the trip-point of the inverter $P_{\mathrm{R}} - N_{\mathrm{R}}(V_{\mathrm{TRIPWR}})$ within the time when word-line is high (T_{WL}). Hence, the write-failure probability (P_{WF}) is given by

$$P_{\mathrm{WF}} = P\left[(T_{\mathrm{WRITE}} > T_{\mathrm{WL}})\right] \tag{5.10}$$

where T_{WRITE} is the time required to pull down V_{L} from V_{DD} to V_{TRIPWR}. T_{WRITE} is obtained by solving:

$$T_{\mathrm{WRITE}} = \begin{cases} \left| \int\limits_{V_{\mathrm{DD}}}^{V_{\mathrm{TRIP}}} \frac{C_{\mathrm{L}}(V_{\mathrm{L}})dV_{\mathrm{L}}}{I_{\mathrm{in(L)}}(V_{\mathrm{L}}) - I_{\mathrm{out(L)}}(V_{\mathrm{L}})} \right| & ;\mathrm{if}(V_{\mathrm{WR}} < V_{\mathrm{TRIPWR}}) \\ \infty & ;\ \mathrm{if}(V_{\mathrm{WR}} \geq V_{\mathrm{TRIPWR}}) \end{cases} \tag{5.11}$$
$$I_{\mathrm{in(L)}} = \text{ current into L } \approx I_{\mathrm{dsPL}}, I_{\mathrm{out(L)}} = \text{ current out of L } \approx I_{\mathrm{dsAX_L}}$$

where C_L is the net capacitance at the node L. V_{WR} can be obtained by solving KCL at nodes L and R [6]. V_{TRIPWR} can be obtained by solving for trip-point of the inverter $P_{\mathrm{R}} - N_{\mathrm{R}}$. In [24], authors have shown that T_{WRITE} obtained using (Equation (5.11)) can closely follow the device simulation result with V_{t} variation of different transistors as shown in Fig. 5.5a. Using (Equation (5.6)), one can estimate the mean $\eta_{\mathrm{TW_R}}$ and the standard deviation ($\sigma_{\mathrm{TW_R}}$) of T_{WRITE} and approximate its PDF as a Gaussian distribution as shown in Fig. 5.5b [24]. However, most of the write failures originate from the “tail” of the distribution function. Hence, to improve the accuracy

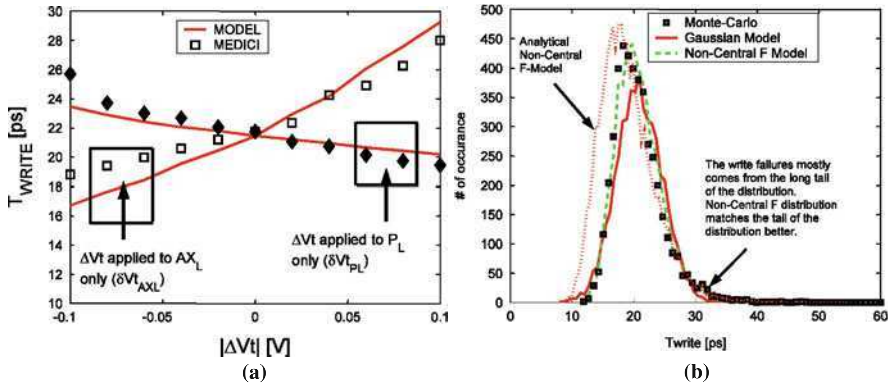


Fig. 5.5 (**a**) T_{WRITE} variation with $\delta\ V_t$ (**b**) distribution of T_{WRITE} [24]

of the model at the tail region, a non-normal distribution can be more accurate. For example, in [24], authors have observed that non-central F distribution may provide closer match to device simulation results [32]. Using the PDF (Gaussian or non-central F) of T_{WRITE}, the P_{WF} can be estimated using (Equation (5.7)).

5.3.2.3 Access Time Failure (AF)

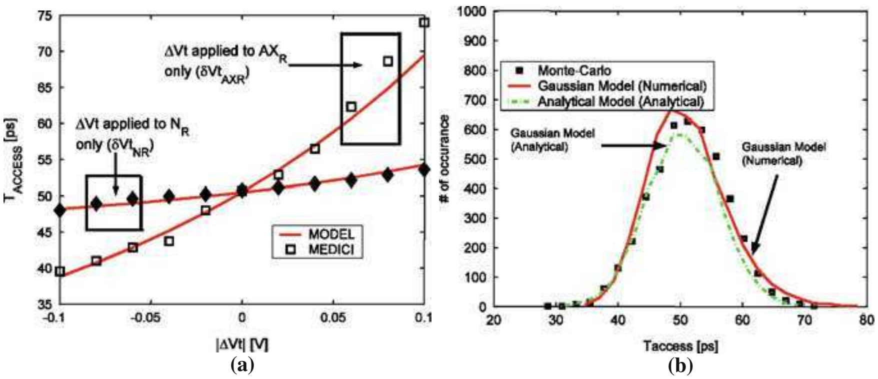
As defined in Section 5.3.2, the access time failure occurs if the access time of the cell (T_{ACCESS}) is longer than the maximum tolerable limit (T_{MAX}). Hence, the probability of access time failure (P_{AF}) of a cell is given by

$$P_{\text{AF}} = P(T_{\text{ACCESS}} > T_{\text{MAX}}) \tag{5.12}$$

While reading the cell storing V_{L}= "1" and V_{R}= "0" (Fig. 5.1b, c), bit-line BR will discharge through AX_{R} and N_{R} by the discharging current, I_{BR}. Simultaneously, BL will discharge by the leakage of AX_{L} of all the cells (I_{BL}) connected to BL in the same column. Hence, the access time is given by

$$T_{\text{ACCESS}} = \frac{C_{\text{BR}} C_{\text{BL}} \Delta_{\text{MIN}}}{C_{\text{BL}} I_{\text{BR}} - C_{\text{BR}} I_{\text{BL}}} = \frac{C_{\text{B}} \Delta_{\text{MIN}}}{I_{\text{dsatAX}_{\text{R}}} - \sum\limits_{i=1,..,N} I_{\text{subAX}_{\text{L}(i)}}} \tag{5.13}$$

where N is the number of cells attached to a bit-line (or column), C_{BR} and C_{BL} are the right and left bit-line capacitances which are assumed to be equal. The access time given by (Equation (5.13)) closely follows the device simulation result as shown in [24] (Fig. 5.6a). The PDF of T_{ACCESS} can be approximated as a Gaussian one with the mean (η_{TAC}) and the standard deviation (σ_{TAC}) obtained from (Equation (5.6)), as shown in Fig. 5.6b. Using the derived PDF ($N_{T\text{ACCESS}}(t_{\text{ACCESS}})$), P_{AF} can be estimated using (Equation (5.7)).



(a) (b)

Fig. 5.6 (**a**) T_{ACCESS} variation with $\delta\,V_t$ (**b**) distribution of T_{ACCESS} [24]

5.3.2.4 Hold Stability Failure (HF)

As discussed in Section 5.3.2, a hold-failure occurs if the minimum supply voltage that can be applied to the cell in the hold-mode (V_{DDHmin}), without destroying the data, is higher than the designed standby mode supply voltage (V_{HOLD}). Thus, the probability of hold-stability failure (P_{HF}) is given by

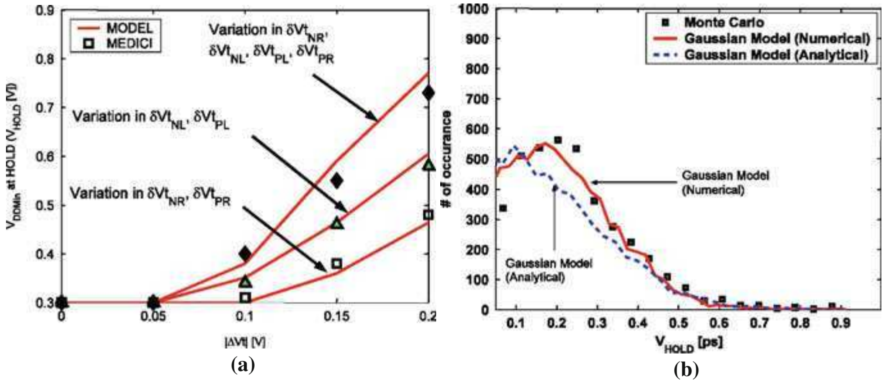
$$P_{\text{HF}} = P\left[V_{\text{DDHmin}} > V_{\text{HOLD}}\right] \tag{5.14}$$

Assume V_{DDH} represents the cell V_{DD} in the hold mode. Lowering the V_{DD} of the cell reduces the voltage at the node storing “1” (V_{L} in Fig. 5.1b). Due to the leakage of N_{L}, V_{L} can be less than V_{DDH} for low V_{DDH}. The hold-failure occurs if V_{L} falls below the trip voltage of the inverter $P_{\text{R}} - N_{\text{R}}(V_{\text{TRIP}})$. Hence, the minimum possible $V_{\text{DDH}}(V_{\text{DDHmin}})$ can be obtained by numerically solving:

$$V_L\left(V_{\text{DDHmin}}, \delta V_{t_{\text{PL}}}, \delta V_{t_{\text{NL}}}\right) = V_{\text{TRIP}}\left(V_{\text{DDHmin}}, \delta V_{t_{\text{PR}}}, \delta V_{t_{\text{NR}}}\right) \tag{5.15}$$

The estimated value of V_{DDHmin} using (Equation (5.15)) can closely follow the values obtained from device simulation as shown in Fig. 5.7a [24]. The distribution of V_{DDHmin} can be approximated as a Gaussian one with mean and variance obtained using (Equation (5.6)), as shown in Fig. 5.7b. Using the Gaussian PDF for V_{DDHmin} the P_{HF} can be estimated using (Equation (5.7)).

The sensitivity-based approach is a fast method of estimating the cell failure probability; however, it is not accurate enough. It is useful for early stage design optimization and exploration. For a more accurate estimation of failure probability that is less complex than standard Monte-Carlo simulation method, the importance sampling Monte-Carlo method can be used [19]. This method is explained in the remaining of this section.



(a) (b)

Fig. 5.7 (**a**) V_{DDHmin} variation with δ V_{t} (**b**) distribution of V_{DDHmin} [24]

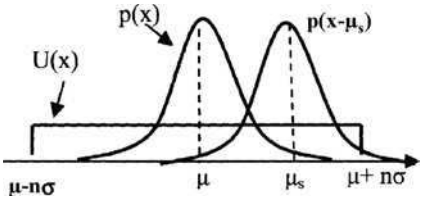
5.3.3 Failure Probability Estimation Using Importance Sampling

Using standard Monte-Carlo simulation to achieve the high accuracy needed for the small cell failure probability requires too many simulations which can be prohibitive in real design scenarios. The issue with standard Monte-Carlo simulation is that most of the samples are around the mean where there are no failures. The importance sampling [12, 16, 18] is a well known technique which overcomes this issue by distorting the sampling function, $p(x)$, to produce more samples in the important region (tail of the distribution). This can be mathematically expressed as:

$$E_{p(x)}\,[\theta] = E_{g(x)}\left[\theta \cdot \frac{p(x)}{g(x)}\right] \tag{5.16}$$

where $g(x)$ is the distorted sampling function. By properly choosing $g(x)$, the accurate results can be achieved with small number of simulations. A typical choice of $g(x)$ can be a uniform distribution function as shown in Fig. 5.8. Another approach is to move original sampling function, $p(x)$, into the failure region by choosing $g(x) = p(x - \mu_S)$ [39]. In [19], authors propose a mixture ratioed importance sampling approach by choosing a mixture of distributions as follows:

Fig. 5.8 Distortion of sampling function from the original normal Gaussian function, $p(x)$, to uniform distribution, $U(x)$, and shifted normal distribution, $p(x\text{-}\mu_S)$



$$g_{\lambda}(x) = \lambda_1 p(x) + \lambda_2 U(x) + (1 - \lambda_1 - \lambda_2)p(x - \mu_s)\;;\, 0 \le \lambda_1 + \lambda_2 < 1 \tag{5.17}$$

This method also enables biasing the distorted sampling function for multiple fault regions and is efficient for multi-dimensional parameter space. In [19], the authors also suggest a heuristic method for selecting the shift, μ_s. This method results in more than 100X speed up compared to the standard Mote-Carlo simulation approach.

5.4 Design Approaches for Variation-Tolerant SRAM

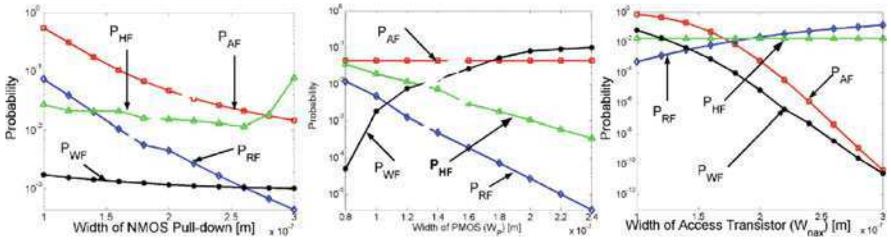
In this section, design techniques for enhancing the variation tolerance of SRAM are discussed. The techniques discussed include (1) optimal SRAM cell sizing and redundancy, (2) dynamic circuit techniques, and (3) adaptive repair techniques.

5.4.1 Optimal Cell Sizing and Redundancy

Transistor sizing in the SRAM cell affects not only the distribution of on-die Vt variations, but also the voltage and timing parameters that determine failures. Hence, optimal transistor sizing in the SRAM cell is an important step in enhancing the memory yield. Let us start by analyzing the sensitivity of different failure probabilities to sizing of different transistors in the SRAM cell.

5.4.1.1 Sensitivity Analysis of Failure Probability

Figure 5.9 shows the dependence of cell failure probabilities to different transistor sizing in the SRAM cell [24]. Figure 5.9c shows that a weak access transistor (small W_{nax}) reduces P_{RF}, which is due to reduction in V_{READ}; however, it increases P_{AF} and P_{WF} and has very small impact on P_{HF}. Reducing the strength of the PMOS pull-up transistors (by decreasing W_{p}) reduces P_{WF} which is due to reduction in the PMOS current (I_{dsPL}), but it increases P_{RF} due to lowering of V_{TRIPRD}. P_{AF} does not depend strongly on PMOS strength (Fig. 5.9b). P_{HF} improves with an increase in W_{p} as the node L get more strongly connected to the supply voltage. Increasing the size of the pull-down NMOS (W_{npd}) increases the strength of pull-down NMOSs (N_{L} and N_{R}). This reduces P_{RF} which is due to reduction in V_{READ}. P_{AF} also reduces by increasing the strength of N_{R} (Fig. 5.9a). Increase in width of N_{R} has little impact on P_{WF}. Although, it slightly increases the nominal value of T_{WRITE}, the reduction of $\sigma_{V_{\mathrm{T}}}$ of N_{R} (see Equation (5.5)) tends to reduce σ_{TWRITE} and hence P_{WF} remains almost constant. A decrease in the V_{TRIP} of $P_{\mathrm{R}}-N_{\mathrm{R}}$ initially reduces P_{HF} with the increase in W_{npd}. However, a higher width of N_{L} reduces V_{L} (from the applied V_{DDH}) due to an increase the leakage of N_{L}. Consequently, a very high W_{npd} increases the P_{HF}. These observations indicate that there are tradeoffs between different failure probabilities, and hence, there is a need for a joint optimization of all transistor sizing to minimize the overall cell failure probability.



(a) Wnpd **(b) Wp** **(c) Wnax**

Fig. 5.9 Variation of failure probabilities with transistor sizing [24]. (**a**) Wnpd (**b**) Wp (**c**) Wnax

Such optimization needs to be constrained by the leakage power and the cell area. Hence, let us first discuss the leakage estimation in SRAM, and then formulate the statistical optimization of an SRAM array.

5.4.1.2 Estimation of Leakage in SRAM

The total leakage of an SRAM cell in bulk silicon technology is composed of the subthreshold, the gate and the junction tunneling leakage [24, 37, 42]. Considering random intra-die variation in V_t, the leakage of different cells (L_{Cell}) in a memory array can be modeled as *independent lognormal random* variables [32]. The mean (μ_{Cell}) and the standard deviation (σ_{Cell}) of the cell leakage can be estimated using Taylor series expansion shown in (Equation (5.6)). The overall leakage of a memory array is the summation of the leakage of all the cells (say N_{cell}) in the array. Using the central limit theorem, the distribution of the overall memory leakage (L_{MEM}) can be assumed to be Gaussian with mean (μ_{MEM}) and the standard deviation (σ_{MEM}) given by [32]:

$$\mu_{MEM} = N_{Cell}\mu_{Cell} \text{ and } \sigma_{MEM} = \sqrt{N_{Cell}}\sigma_{Cell} \tag{5.18}$$

Hence, the probability that L_{MEM} is less than a maximum allowable limit (L_{MAX}) is given by

$$P_{LeakMEM} = P\left[L_{MEM} < L_{MAX}\right] = \Phi\left(\frac{L_{MAX} - \mu_{MEM}}{\sigma_{MEM}}\right) \tag{5.19}$$

SRAM chips that have too much leakage (more than L_{MAX}) are considered to be faulty due to excessive power consumption. $P_{LeakMEM}$ is an indicator of the power yield of the SRAM. In optimization of SRAM array, $P_{LeakMEM}$, can be constrained to be greater than a minimum target value ($P_{LeakMin}$).

5.4.1.3 Statistical Optimization of SRAM

A statistical optimization problem for design of SRAM array can be stated as

$$\text{Minimize } P_{\text{MEN}} = f(\mathbf{X})$$
$$\text{where } \mathbf{X} \equiv [L_{\text{nax}},\ W_{\text{nax}},\ L_{\text{npd}},\ W_{\text{npd}},\ L_{\text{pup}},\ N_{\text{RC}}]$$

Subject to:

$$P_{\text{LeakMean}} \geq P_{\text{LeakMin}}$$
$$A_{\text{MEM}} \leq \text{Maximum Area}\,(A_{\text{MAX}})$$
$$E[T_{\text{AC}}] = \mu_{\text{TACCESS}} \leq \text{Maximum access time mean}\ (\mu_{\text{TAC - MAX}})$$
$$\text{For all the parameters: } \{X_{\text{MIN}}\} \leq \{X\} \leq \{X_{\text{MAX}}\}$$
$$A_{\text{MEM}} = A_{\text{actual}} + A_{\text{redundant}} = N_{\text{ROW}}(N_{\text{COL}} + N_{\text{RC}})\,A_{\text{cell}}$$

where A_{MEM} is the total memory area and A_{cell} is the area of the memory cell. This is essentially a non-linear optimization problem with non-linear constraints. The upper bound on the mean access time is given to ensure that robustness of the memory has not been achieved by significantly sacrificing performance. The upper bound of N_{RC} is determined using (Equation (5.20)) as shown below:

$$N_{\text{RCmax}} = \frac{A_{\text{MEM}} - N_{\text{ROW}} N_{\text{COL}} A_{\text{cellmin}}}{A_{\text{cellmin}} N_{\text{ROW}}} = \frac{A_{\text{MEM}}}{A_{\text{cellmin}} N_{\text{ROW}}} - N_{\text{COL}} \tag{5.21}$$

The minimization of P_{F} in the optimization procedure requires the estimation of the joint probabilities as expressed in (Equation (5.4)) which are computationally expensive. However, it should be noted that:

$$P_{\text{F}} = P\,[A_{\text{F}} + R_{\text{F}} + W_{\text{F}} + H_{\text{F}}] \leq P_{\text{AF}} + P_{\text{RF}} + P_{\text{WF}} + P_{\text{HF}} = P_{\text{FMOD}} \tag{5.22}$$

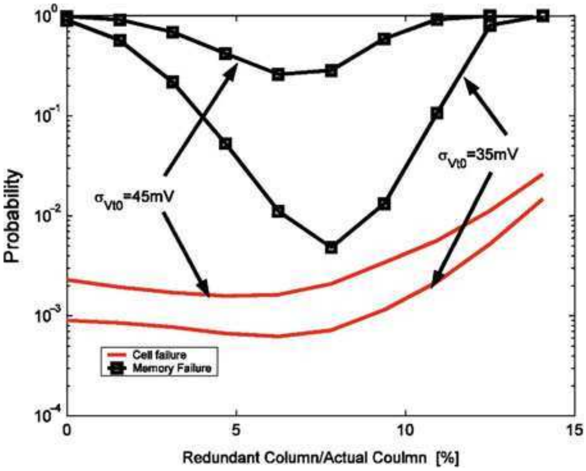
Hence, instead of minimizing P_{F} , P_{FMOD} can be minimized. The above problem can be solved using *Lagrange Multiplier* based algorithm [24, 44].

In [24], authors have applied the optimization methodology described earlier to optimize the cell structure obtained from [1]. The results of the optimization are shown in Table 5.1. To improve the beta ratio between the pull-up PMOS and access transistor, the original cell was designed with a longer PMOS. However, a weaker PMOS tends to increase the read failure. Hence, the optimization reduces the length of the PMOS and uses the extra area in the pull-down NMOS, thereby reducing the access failure probability. The statistical optimization algorithm also allowed to trade-off between the redundancy area and the active cell area. Reducing the number of redundant columns allows more area for each of the actual cells. This reduces the failure probability of the cells, thereby reducing P_{MEM}. On the other hand, from

Table 5.1 Optimization results [24]

	$\beta_{\text{nax}}/\beta_{\text{p}}$	$\beta_{\text{npd}}/\beta_{\text{nax}}$	P_{F}	P_{MEM}	I_{Leak}	T_{AC}	Yield(%)
Original cell [1]	1.5	1.36	2.6e-3	0.034	851 µA	55 ps	47
Optimized cell	1.2	1.48	3.4e-5	0.001	950 µA	46 ps	95

Fig. 5.10 Impact N_{RC} on memory yield [24]



(Equation (5.2)) it can be observed that reducing N_{RC} will tend to increase P_{MEM}. Figure 5.10 shows the variation of P_{MEM} with the variation of N_{RC} considering constant A_{MEM}. It can be observed that increasing the redundancy beyond a certain point increases the memory failure probability. It should be further noted that with the application of a higher value of the $\sigma_{\mathrm{V_{t_0}}}$ ($\sigma_{V_{\mathrm{t}}}$of a minimum sized transistor following (Equation (5.5))), the optimized value of the redundancy (that minimizes failure probability) reduces. This indicates that with larger amount of variations, design of robust cell is more effective in reducing the failure probability (improving yield) as compared to increasing number of redundant columns. Hence, it can be concluded that a statistical analysis of effectiveness of the redundancy is necessary to improve the memory failure probability.

The static noise margin (SNM) of a cell is often used as a measure of the robustness of an SRAM cell against flipping [2]. However, an increase in SNM makes the cell difficult to write by increasing its data holding capability, which increases write failures as shown in Fig. 5.11. Consequently, an increase in the SNM does not necessarily reduce the overall failure probability. In other words, maximizing SNM alone does not guarantee maximum yield; however, using the discussed statistical optimization method based on failure probabilities targets maximizing the memory yield.

5.4.2 Dynamic Circuit Techniques

As observed from Fig. 5.9, conflicting requirements imposed by read and write operations and cell disturbs make it difficult to simultaneously mitigate the various failure mechanisms. Changing a transistor size reduces one failure probability while it may increase another failure probability. Overcoming this limitation requires

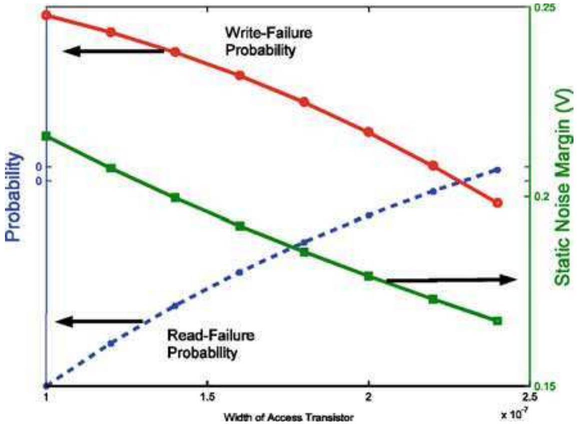


Fig. 5.11 Impact of SNM on failure probability of SRAM cell [24]

innovative circuit techniques to reduce one or more type of failures with minimal increase in other failure mechanisms. Dynamic modifications of cell terminal voltages, depending on the type of operation, provide more flexible ways to simultaneously reduce all failure types.

Write-ability can be improved by dynamically altering the word-line or cell supply voltage using row-based or column-based control [3, 7, 17, 31, 51]. A higher word-line voltage (stronger access transistors), or reduced cell-supply voltage (weaker pull-up devices) during write operation improves cell write-ability [3, 7]. However, row-based dynamic control of terminal voltages during write increases read disturb failures in the *half-selected* cells (in unselected columns) along the selected row. This effect can be avoided using column-based dynamic control of the cell supply voltage [31, 51]. In this case, the supply voltage of the selected columns is reduced during write but that of unselected columns is kept at nominal voltage. Figure 5.12 shows the block diagram of SRAM subarray with integrated column-based dynamic supply voltage [51]. Low supply voltage is applied only to

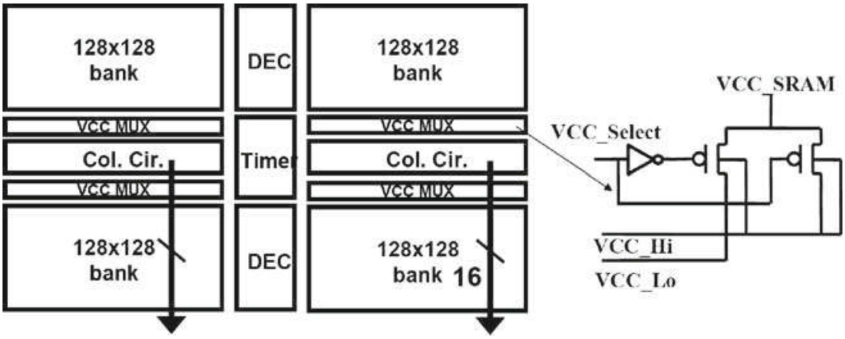


Fig. 5.12 SRAM with integrated column-based dynamic power supply [51]

the selected columns during the write operation, improving the write-ability. During the read operation, the high supply voltage is selected, improving the read stability. For unselected banks, low supply voltage is applied to reduce the leakage power. An alternative scheme is proposed in [31], where there is no need for a low supply voltage on-chip. Instead the supply voltage of the selected column is lowered by isolating its supply line and connecting it to an already discharged dummy supply line. In this way, the supply voltage is lowered by charge sharing between the isolated column supply line and the dummy supply line.

However, a lower supply voltage during write operation also reduces the trip-point of the inverter PR–NR and gain of the cross-coupled inverter pair which negatively impact the cell write-ability. In [28], authors propose a technique for reducing write failures by using a transient negative voltage at the low-going bit-line during the write cycle (Fig. 5.13). The authors also propose a capacitive-coupling based transient negative bit-line voltage scheme to eliminate the need for on-chip generation of a small negative voltage (Fig. 5.13). This preserves the column based control, and enhances the strength of the access transistor with no change in the cross-coupled inverter. Enhancing the strength of the access transistor improves the write ability. The design includes two boosting capacitors, connected to bit-lines BL and BR. The capacitors can be implemented using MOSFET gate capacitances. The other end of the capacitors is connected to the BIT_EN signal. The signal NSEL is used for column selection. BIT_EN and NSEL are synchronized with word-line signal, read/write (WR) signal, and column select (CS) signals. The NSEL and BIT_EN signals are asserted along with the WL pulse, and de-asserted midway through the WL pulse. This turns off the pass transistors N_{BL}/P_{BL} and N_{BR}/P_{BR}, leaving bit-lines BL and BR floating at 0 and 1, respectively. Next, the high-to-low

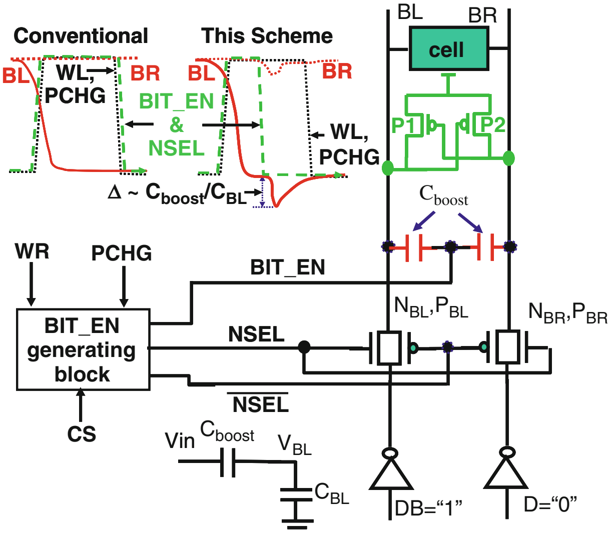


Fig. 5.13 SRAM with transient negative bit-line voltage to improve write ability [28]

transition of BIT_EN causes an under-shoot at the bit-lines BL and BR due to capacitive coupling through the boosting capacitors. As the bit-line BL is floating at 0, this causes a temporary negative voltage at bit-line BL. The transient negative voltage at BL results in a transient increase in the discharging current of the access transistor SL, thus facilitating the pull-down of the node L voltage (V_L). Simulation results in a 45-nm PD/SOI technology show a 10^3X reduction in write failure probability and marginal reductions in access and read disturb failures at a small (7–9%) area overhead [28].

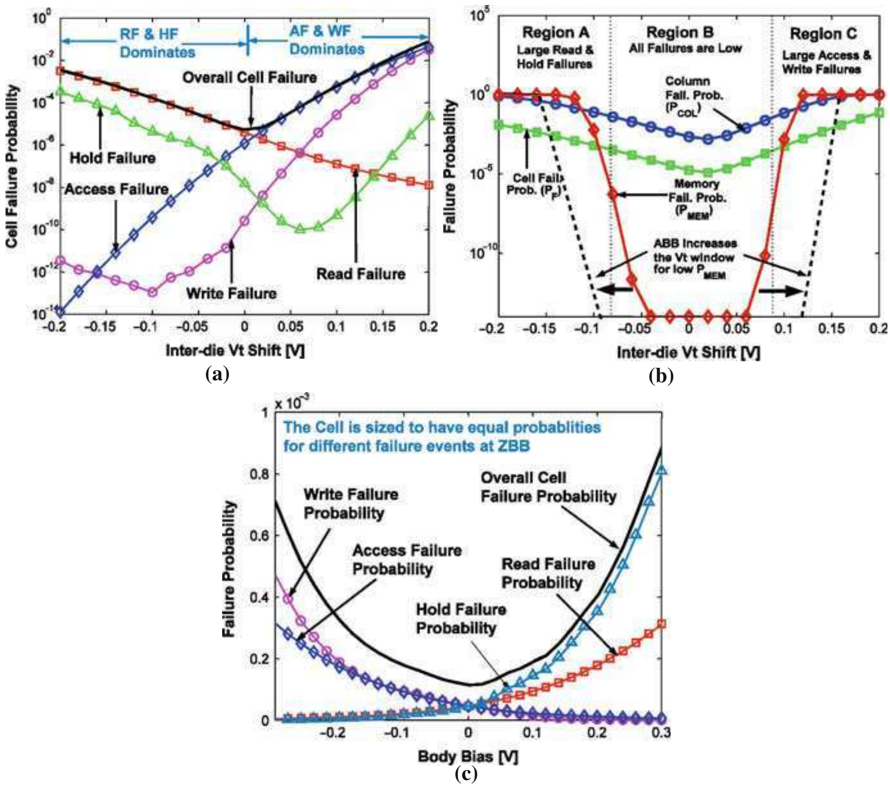
5.4.3 Adaptive Repair of SRAM

As discussed in Section 5.3.3, die-to-die and within-die statistical variations in process parameters result in parametric failures in SRAM cell, thereby degrading design yield. The principal reason for parametric failures is the intra-die variation in threshold voltage of the cell transistors due to Random Dopant Fluctuations (RDF) [2, 24]. On the other hand, the die-to-die variation in process parameters (say, V_t) can significantly enhance the impact of within-die random variations [26, 27]. The pre-silicon design/optimization approaches discussed in previous Sections (5.3.4.1 and 5.3.4.2) help to improve the tolerance to random process variations, but often fail short if die-to-die variations are non-negligible. The post-silicon adaptive repair (along with pre-silicon design techniques) provides a new opportunity to improve SRAM yield considering both local and global variations [26, 27].

It is evident from Section 5.3.3 that the primary cause of parametric failures in SRAM array is local device variations which creates a large number of faulty cells in the array. Due to this local nature of the faults, it is impossible to correct each and every faulty cell in a faulty array. Redundancy can help to mask the effects of a certain number of faults. But as mentioned earlier (see Fig. 5.10), too much redundancy for a constant array area can negatively impact cell failure and memory failure [24]. The aim of the post-silicon adaptive repair of a faulty SRAM array is to reduce the total number faults to a level such that they can be corrected using available redundancy. The adaptive repair approach is based on *the analysis and reduction of the effect of die-to-die variations to reduce the total number of faults in an array* [26, 27]. Let us assume that the SRAM cell and redundancy are optimized such that the memory failure probability due to local random variations is negligible at nominal inter-die corner. The existing pre-silicon design techniques (discussed in Sections 5.3.4.1 and 3.4.2) can be used to achieve this goal.

5.4.3.1 Effect of Die-to-Die Variation on SRAM Failures

In [26] and [27], authors have analyzed the effect of global die-to-die variations on the different types of failures. A negative inter-die V_t shift (i.e., for the SRAM arrays shifted to the low-V_t process corners) increases the read and the hold failures as illustrated in Fig. 5.14a. This is because of the fact that lowering the V_t of the cell transistors increases V_{READ} and reduces V_{TRIPRD}, thereby increasing read failure



(a) (b) (c)

Fig. 5.14 Effect of inter-die V_t shift and body-bias on the failure probabilities: (**a**) cell failure probability with inter-die V_t shift, (**b**) memory failure probability with inter-die V_t shift, and (**c**) effect of body-bias on cell failure [27]

probability. The negative V_t shift increases the leakage through the transistor N_L, thereby, increasing the hold failure probability. In case of the SRAM arrays in the high-V_t process corners, the access time failures and the write failures are high as illustrated in Fig. 5.14a. This is principally due to the reduction in the current drive of the access transistors at high V_t corners. The hold failure also increases at the high V_t corners, as the trip-point of the inverter PR–NR increases with positive V_t shift. Hence, the overall cell failure increases both at low and high-V_t corners and is minimum for arrays in the nominal corner as shown in Fig. 5.14a. Consequently, the probability of memory failure is high at both low-V_t and high-V_t inter-die process corners as shown in Fig. 5.14b.

5.4.3.2 Effect of Body-Bias on Cell Failures

Since body biasing can change V_t of the whole die, the effect of the body-bias can be investigated on memory failure probabilities. In [26] and [27], authors have investigated the effect of NMOS body bias on different types of failures to develop an adaptive repair technique. Application of reverse body-bias increases V_t of transistors which reduces V_{READ} and increases V_{TRIPRD}, resulting in a reduction in the read failure as shown in Fig. 5.14c. The V_t increase due to Reverse Body Bias (RBB) also reduces the leakage through the NMOS thereby reducing hold failures (Fig. 5.14c). However, increase in the V_t of the access transistors due to RBB increases the access and the write failures. On the other hand, application of Forward Body Bias (FBB) reduces the V_t of the access transistor which reduces both access time and write failures as seen in Fig. 5.14c. However, it increases the read failure because V_{READ} increase and V_{TRIPRD} reduces. FBB also increases hold failures due to increased leakage through NMOS.

5.4.3.3 Adaptive Self-Repair of SRAM Array

As observed in the previous section, in each corner of an inter-die variation spectrum, there are dominating failure type(s). Hence, if one could detect the inter-die (global) variation corners and/or the dominant type of failure(s), then it is possible to reduce the total number of failures by adaptively changing the biasing of the memory to reduce the dominant failure(s). Notice that it is sufficient to reduce the dominant types of failures in each corner, in order to reduce the overall cell failure probability. Based on this approach, an adaptive self-repair method can be developed to let the chip adjust its own biasing in order to minimize the number of faulty cells, or in another words, to repair the cell that would otherwise fail. Hence, this approach is a post-silicon yield enhancement strategy and requires two basic design elements. One is a circuitry for detection of the global variations corner and the other one is a circuitry for adjusting SRAM biasing.

In [25], [26] and [27], authors have demonstrated an adaptive self-repair SRAM. Their design monitors the leakage of the total SRAM array to determine the inter-die corner as shown in Fig. 5.15a[26]. Based on the sensed leakage, a forward, reverse, or zero body-bias can be applied to reduce the total number of failures. As the memory leakage is summation of the leakages of a large number of cells, the local random variations in the cell leakage do not increase the spread of the memory leakage [26, 32, 36]. Consequently, the SRAM array leakage is a good indicator of the inter-die corner. Using the statistical design optimization discussed in Section 5.3.4.1, the size of the transistors in the SRAM cell is first optimized to minimize the cell failure probability at nominal inter-die corner. Without adaptive repair the designed SRAM array has large number of failures in low-V_t and high-V_t inter-die corners resulting in a low yield, particularly, for large inter-die variations as shown in Fig. 5.15b [26, simulations in predictive 70-nm node]. From Fig. 5.14b, it can be observed that, above a certain V_t-shift (~100 mV), small changes in inter-die V_t results in a large memory failure probability (~1) (regions A and C). However, for chips with V_t in the window of –100 mV to 100 mV (region B) the memory failure probability (P_{MEM}) is negligible (~0). Hence, to improve yield, the number of dies in region A and C need to be reduced. This is achieved by applying Reverse Body Bias (RBB) to the dies in region A thereby reducing their read and hold failure probability. Similarly, application of Forward Body Bias (FBB) to the chips in

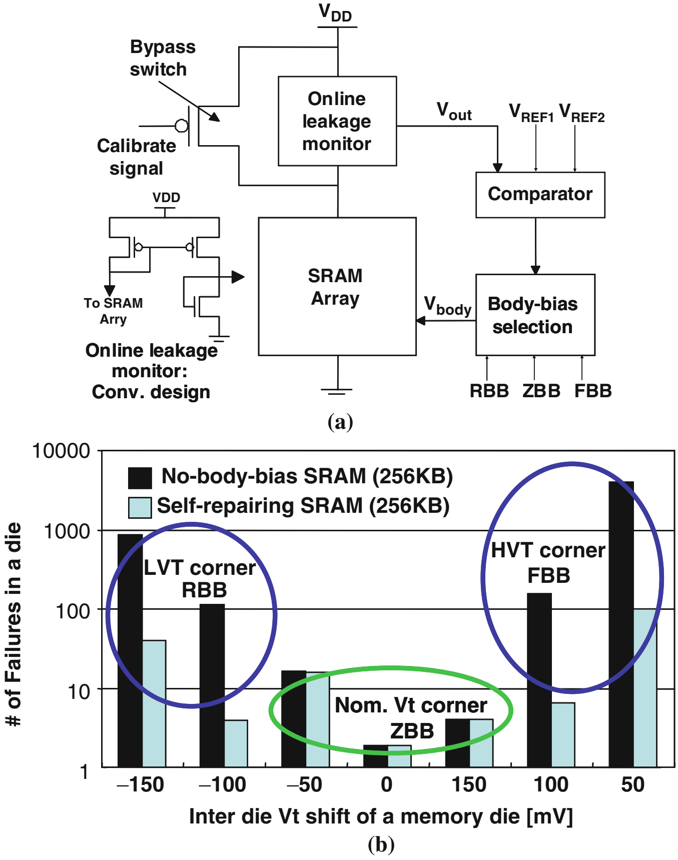


Fig. 5.15 Adaptive repair of SRAM: (**a**) Self-repairing array and (**b**) reduction in number of failures in 256 KB self-repairing array [26]

region C reduces their write and access time failure probability. Application of RBB in low-V_t corner and FBB in high-V_t corner results in a large reduction in number of failures in both low and high inter-die V_t corners. In the design shown in Fig. 5.15a, the forward and reverse body bias voltages are ±0.3 V. In [26], authors also demonstrated a test-chip designed in 130-nm CMOS to verify the effectiveness of adaptive repair of SRAM.

The discussed adaptive repair technique assumes that the global die-to-die variations in PMOS and NMOS devices are always correlated. This assumption is not always valid. The uncorrelated PMOS and NMOS variations impact the dependence of parametric failures on global corners. For example, the write failures are maximum at low-V_t PMOS and high-V_t NMOS corners, while read failures are minimum in this corner. Similarly, at high-V_t PMOS and low-V_t NMOS corners the read failures are maximum, while write failures are minimal. Hence, more efficient repair methods need to adapt to global corners of both NMOS and PMOS.

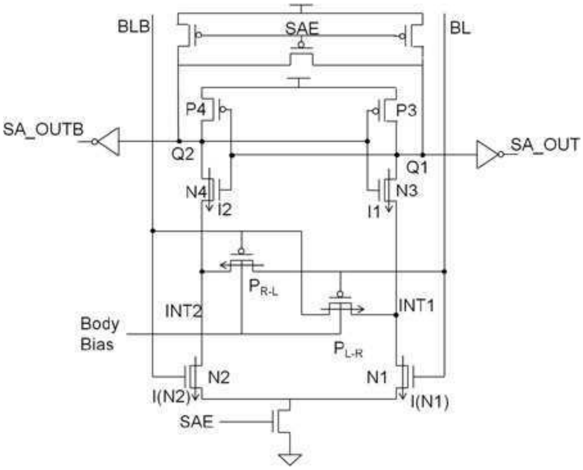
For example, Yamaoka et al. have proposed techniques to separately measure the PMOS and NMOS leakage in SRAM arrays to determine their individual global corners [47]. The separate NMOS and PMOS body-bias was considered to repair their global variations and reduce dominant types of failures under all global corners. Mojumder et al. have proposed a different approach for implementation of adaptive repair [22]. Instead of identifying the global V_t corners, an SRAM-based test-circuit using large transistors (or a set of parallel transistors) are used to directly measure the global read and write margin corners for every die. Further, instead of using body-bias, variable cell supply and word-line voltage are used to perform adaptive repair. For dies shifted to worse global read margin corner, a higher cell supply and lower word-line voltage are used (∼100–200 mV voltage difference is sufficient). For the dies shifted to worse global write margin corners, lower cell supply and higher word-line voltages are used. As the adaptation is performed using voltage of the cell terminals, this method is applicable to both bulk-CMOS and SOI technologies.

5.4.4 Variation Tolerance for SRAM Peripherals

The main peripherals surrounding the SRAM array include the row address decoder, column multiplexer, and the sense-amplifier as shown in Fig. 5.1a. The read and write operation involve use of the peripherals, and hence, the SRAM failures may be coupled with the process variations in the peripheral circuits. The variations can also exist in the bit-line and word-line capacitances and resistances due to variations in interconnect. In particular, the read access failure in an SRAM array can occur due to random V_t variation in SRAM cell or sense-amplifier [10, 29]. Hence, analysis of access failure has to consider the impact of both the SRAM cell and sense-amplifier. Sense amplifiers rely on differential pair transistors to detect the small voltage differential on the input bit-lines [35]. For a sense amplifier to operate correctly, the voltage differential between the input bit-lines must be greater that a minimum values, called the offset voltage, otherwise the sense amplifier will produce wrong data at its output. Intra-die process variations cause mismatch in the differential pair transistors of the sense amplifier, and hence, a non-zero offset voltage for the sense amplifier [29]. The sense amplifier offset voltage also varies from column to column. Process variations in the SRAM array cause the bit-line voltage differential to vary from cell to cell. Probability of read access failure can be quantified as the probability that the voltage differential established on the bit-lines during the read cycle falls below the offset voltage of the sense amplifier [29]. Improving the robustness of sense-amplifier can significantly reduce the access failure probability in memory.

The V_t (similarly, L and W) mismatch in the transistors has a considerable impact on the stability of sense amplifiers [33, 38, 46]. DRAM designers have proposed several offset compensation schemes [20,33, 40]. These techniques try to compensate the mismatch by either precharging the bit-lines or the two sense amplifier outputs to different initial values. However, in SRAM, bit-lines and the outputs are precharged

Fig. 5.16 Sense amplifier with PMOS stabilizer [29]



and equalized to V_{DD} [35]. Hence, these techniques cannot be directly applied to SRAM sense amplifiers.

To improve the process variation tolerance in sense-amplifier, authors in [29] have proposed a feed-forward path based stabilization scheme, implemented using leakage current, as shown in Fig. 5.16. They have introduced two PMOS stabilizer transistors, P_{L-R} and P_{R-L}, between the inputs (i.e., the bit-lines BL and BLB) and the drain of the differential pair transistors (N1 and N2). The PMOS stabilizers provide a feed-forward mechanism where input bit differential produces an additional discharging current difference at the output. Hence, for a correct sensing operation, the feed-forward scheme requires a lower input voltage difference between the bit-lines BL and BLB, compared to the original design. In other words, this scheme reduces the input offset voltage, thereby reducing the probability of the incorrect operation.

In [49], authors have demonstrated a statistical transistor sizing and dual- V_t optimization for sense-amplifier circuit to reduce its failure probability. They show that assigning high V_t to differential pair transistors (N1 and N2 in Fig. 5.16) and low V_t to other transistors significantly reduces the failure probability of the sense amplifier. In [10], authors have proposed the idea of double sensing which is to have two parallel sense amplifiers to sample the bit-lines twice during one read cycle. The second sense amplifier is fired in the same read cycle but with a slight delay compared to the first sense amplifier. Delayed firing of the second sense amplifier allows more time for the cell to establish a larger voltage differential on the bit-lines, and hence, reduces the possibility of failure on the second sense amplifier. Double sensing can be used during SRAM test mode for detection of read faults and during normal operation for online detection and correction of any number of random read faults [10]. The outputs of the two sense amplifiers are compared and if a mismatch is detected, it is an indication of a failure on the first sense amplifier, and hence, it is corrected by the value sampled by the second sense amplifier.

5.5 Low-Power SRAM Under Process Variation

Forming a yield perspective, the increasing variation manifests itself as yield degradation. On the other hand, variation is also a critical bottleneck for power reduction. Reduction of array operating voltage can significantly reduce the SRAM power. However, a lower supply voltage increases the sensitivity of SRAM to manufacturing variations, and hence, the probabilities of parametric failures increase with voltage scaling [2, 17, 24, 27]. An SRAM cell which is functional at the nominal supply voltage can fail at a lower voltage. From a system perspective this leads to a higher bit-error rate with voltage scaling and limits the opportunity for power saving.

This section discusses post-silicon adaptation methods that improve SRAM yield and reduce power dissipation under process variation in the context of on-chip SRAM for multimedia systems. The yield of on-chip SRAM can significantly impact the cost of multimedia systems. Further, as the SRAM power is a significant component of the system power consumption in multimedia systems, power reduction in SRAM is important [9, 11, 13, 21, 48, 50].

In multimedia systems, the SRAM array is shared between data storage and image storage applications. The data processing applications cannot tolerate any memory failures. Post-silicon repair of SRAM arrays can be done to ensure correct functionality at nominal supply voltage. The different types of parametric failures can increase at different inter-die corners. As discussed in Section 5.3.4.3, the post-silicon repair in SRAM array aims to reduce the dominant types of failures at different inter-die corners using adaptive biasing thereby correcting faulty arrays and improving design yield. With such adaptive repair, the SRAM arrays operate in “always correct” mode at nominal supply voltages.

The increase in the parametric failures at lower supply voltage limits the voltage, and hence, power reductions in SRAM [11, 13, 21, 50]. Although, this is a serious bottleneck when SRAM arrays store data, image processing and multimedia applications can provide acceptable quality-of-service even with a non-negligible amount of bit-error rate [11, 13, 21, 50]. Aggressive voltage scaling can be performed in SRAM arrays for multimedia applications exploiting this inherent error tolerance [11, 13, 21, 50]. However, aggressive scaling of voltage in all bits of a pixel (uniform voltage scaling) increases the error rates in all bits and leads to faster degradation of image quality with voltage scaling. To counter this effect recently, spatial voltage scaling has received significant attention. In [8, 15], authors have explored the power advantage of using different voltages across a word for arithmetic circuits. In a recent study, Yi et al. have shown that such a concept can also be very useful for improving effective yield of video memories [50]. Through a system-level experimental analysis with varying bit-error-rate of the memory, the authors have shown that the higher voltage for Higher-Order Bits (HOB) and nominal voltage for Lower-Order Bits (LOB) can help improve yield of SRAM for multimedia applications.

Cho et al. have presented a reconfigurable SRAM architecture that exploit this property and provide the necessary accuracy during data storage while performing accuracy-energy trade-off when storing image [11]. The basic approach is to dynamically adapt the voltage levels of bits as well as number of bits in the low-voltage modes. The overall SRAM architecture to implement the run-time adaptive accuracy-energy trade-off is shown in Fig. 5.17 [11]. The conventional SRAM array with column multiplexing is considered where same-order bits of different words are grouped together (hereafter referred to as MUX group). A single read/write circuit is used per MUX group. Therefore, an entire MUX groups need to be at a same voltage level. The key requirement of this approach is that, the cell supply, bit-line precharge, write voltage (i.e., the voltage applied to bit-line of logic "1" while writing) need to be at the same voltage. The supply lines of the cells in a MUX group are connected together. The supply networks for different MUX groups are disconnected to allow bit-by-bit reconfiguration. The cell supply of a MUX group is also connected to the corresponding precharge device and the write driver supply. Changing the precharge voltage using a voltage switching network (the top PMOSes in Fig. 5.17) is sufficient to change the cell supply, bit-line, and write voltage for the array.

A reconfigurable Word-Line (WL) structure is used as shown in Fig. 5.17. In this structure, the local WLs of a row of cells in a MUX group are connected together. The local WLs of different MUX groups are disconnected. Global WL is connected

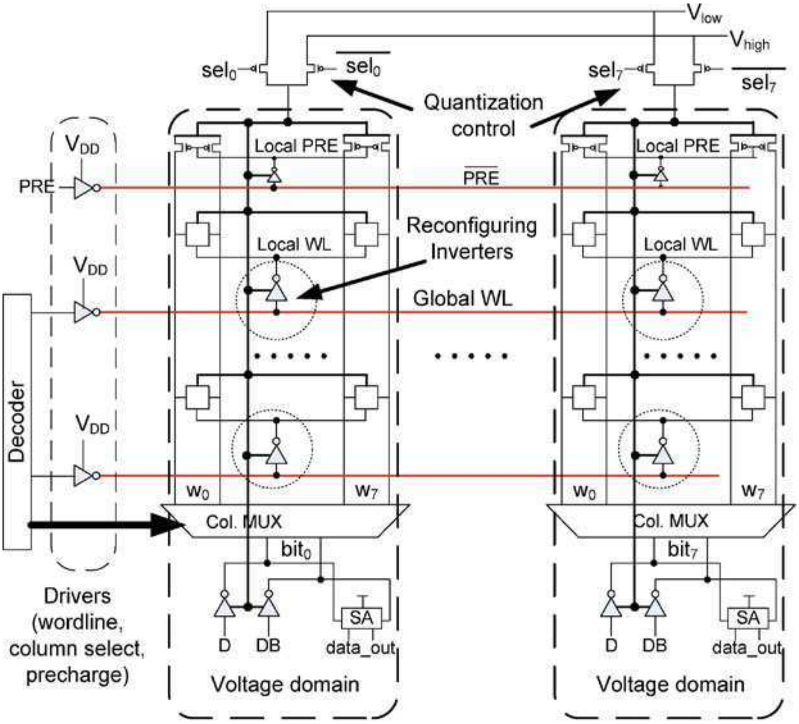


Fig. 5.17 Adaptive accuracy-energy trade-off in SRAM [11]

to reconfiguring inverters to produce local WL signals. The reconfiguring inverters are connected to the supply voltage of their own MUX group, and hence, they act at voltage level shifters. The voltage reconfiguration network consists of two PMOSes – one connected to the low voltage and the other to the high voltage. The gate voltage of the PMOSes for the *j*th bit is controlled by the *j*th select signal (sel_j). To reconfigure a bit to the low voltage, sel_j is set to “0”. Note this architecture also allows uniform voltage scaling which can be performed by setting sel_j=0 for all the bits and changing the voltage to V_{low}.

Figure 5.18 shows that significant power saving can be obtained with a graceful degradation of image quality for different images [11]. The image quality degradation is estimated by comparing the Mean Structural Similarity (MSSIM) index of the original and modified image [45]. If two images are visually identical, MSSIM is 1. A degradation in the image quality results in a lower MSSIM. As observed from Fig. 5.18, number of LOBs (L_{bit}) = 8 provides more power saving at the cost of quality. Using $L_{\text{bit}} = 4$, approximately 45% power saving can be obtained compared to a regular array with a 10% reduction in quality. Power saving is 20% less than the saving achievable by reducing voltage of all bits at the same degradation level. Increasing the number of low-voltage bits degrades the quality with an increase in power saving over the regular array. The quality degrades slowly till sixth

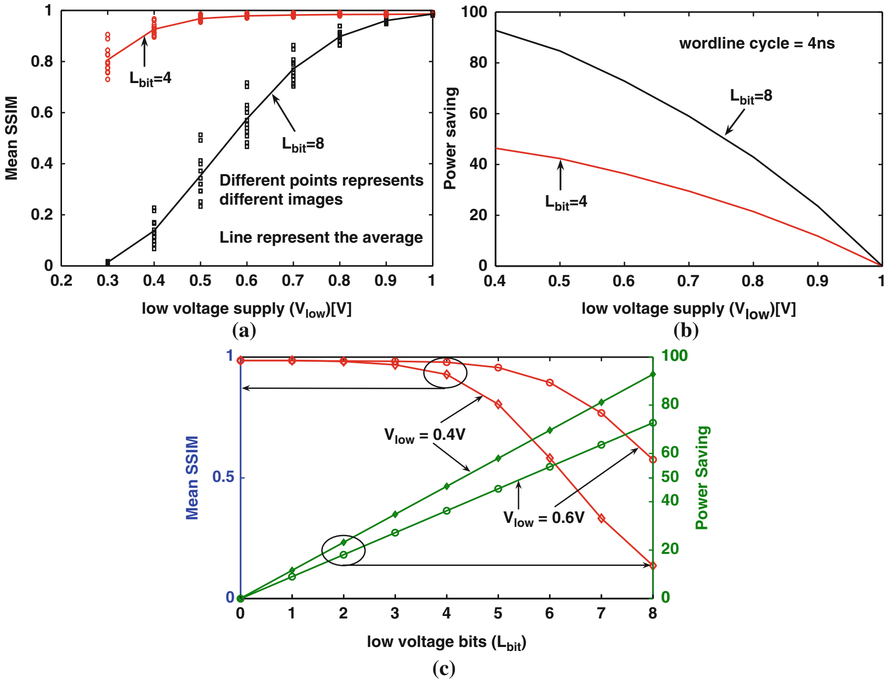


Fig. 5.18 Effect of supply voltage scaling on (**a**) image quality and (**b**) power; (**c**) effect of number of low order bits (L_{bit}) [11]

bit, but reconfiguring the seventh and eighth bit to low voltage mode can result in significant error. Further, a higher V_{low} provides a higher room for re-configuration with a lower power saving in each configuration. If V_{low} and L_{bit} are reconfigured simultaneously, better energy-accuracy trade-off and higher power saving can be obtained. However, as fine-grain change in the supply voltage increases the design complexity, a more practical approach is to select a low voltage level and use L_{bit} for reconfiguration.

5.6 Summary and Conclusions

Technology scaling and increasing intra-die parameter variations increase parametric failures in SRAM. This chapter has discussed different parametric failure mechanisms that can occur in SRAM cell in nano-scale CMOS. A model has been discussed to predict the parametric yield of an SRAM array in terms of cell failure probability. To estimate SRAM cell failure probability due to parameter variations, both numerical Monte-Carlo simulation based methods and a semi-analytical method have been discussed. The standard Monte-Carlo simulation method is the most accurate, but most time consuming method for estimating the cell failure probability. The semi-analytical method using the sensitivity-based approach is simple but inaccurate, which is suitable for early design exploration. The importance sampling Monte-Carlo method is almost as accurate as standard Monte-Carlo, but uses less number of samples by properly biasing the samples to failure regions.

Design methods for enhancing variation tolerance of SRAM have been discussed. The statistical optimization of the SRAM array has been formulated to maximize its yield by joint optimization of the sizing of the transistors in the SRAM cell and column redundancy in the array. Given the trade-off between different failure probabilities, design optimization alone may not be sufficient to achieve acceptable yield under large process variations. For further enhancement of the yield, the chapter has summarized the dynamic circuit design techniques that dynamically change the biasing of cell between the read and write operation modes to reduce both the read and write failure probabilities. The effect of inter-die variation on different parametric failures in memory has been analyzed. The analysis shows that read/hold failures are higher in SRAM dies shifted to low inter-die V_{t} corners, whereas write/access failures are higher in dies shifted to high- V_{t} corners. Adaptive techniques can be applied to adjust cell voltages depending on the inter-die process corner to reduce the dominant failure, and hence, the overall failure rate. The chapter has discussed a self repairing SRAM that uses adaptive body bias for post-silicon adaptive repair of SRAM array, resulting in better memory yield. Variation tolerance for SRAM peripherals and yield enhancement methods for sense amplifiers have been discussed. Finally, the chapter has discussed an adaptive voltage scaling method to offer a trade-off between variation tolerance and low power in multimedia applications, where some level of errors can be tolerated by the application.

References

1. Agarwal A, Hai Li, Roy K (Feb 2003) A single-Vt low-leakage gated-ground cache for deep submicron. IEEE J Solid-State Circuits 38(2):319–328
2. Bhavnagarwala A, Tang X, Meindl JD (Apr 2001) The impact of intrinsic device fluctuations on CMOS SRAM cell stability. IEEE J Solid-State Circuits 36:658–665
3. Bhavnagarwala A, Kosonocky SV, Kowalczyk SP, Joshi RV, Chan YH, Srinivasan U, Wadhwa JK (Jun 2004) A transregional CMOS SRAM with single, logic VDD and dynamic power rails. In: Symposium on VLSI Circuits, Honolulu, HI, pp 292–293
4. Borkar S, Karnik T, Narendra S, Tschanz J, Keshavarzi A, De V (2003) Parameter variation and impact on circuits and microarchitecture. Design Automation Conference, Anaheim, CA, pp 338–342
5. Burnett D, Erington K, Subramanian C, Baker K (Jun 1994) Implications of fundamental threshold voltage variations for high-density SRAM and logic circuits. In: Symposium on VLSI Technology, Honolulu, HI, pp 15–16
6. Chandrakasan A, Bowhill WJ, Fox F (2001) Design of high-performance microprocessor circuits. IEEE, Piscataway, NJ
7. Chappell B, Young I (Apr 2003) VDD modulated SRAM for highly scaled, high performance cache. US Patent 6556471
8. Cheemalavagu S, Korkmaz P, Palem KV (Sept 2004) Ultra low-energy computing via probabilistic algorithms and devices: CMOS device primitives and the energy-probability relationship. In; International conference on solid state devices and materials, Tokyo, pp 402–403
9. Chen G, Kandemir M (2005) Optimizing address code generation for array-intensive DSP applications. In: International symposium on code generation and optimization, San Jose, CA, pp 141–152
10. Chen Q, Mahmoodi H, Bhunia S, Roy K (Nov 2005) Efficient testing of SRAM with optimized march sequences and a novel DFT technique for emerging failures due to process variations. IEEE Trans VLSI Syst 13(11):1286–1295
11. Cho M, Schlessman J, Wolf W, Mukhopadhyay S (2009) Accuracy-aware SRAM: a reconfigurable low power SRAM architecture for mobile multimedia applications. In: Asia and South Pacific design automation conference, Yokohama, pp 823–828
12. Cochran WG (1977) Sampling techniques, 3rd edn. Wiley, New York, NY
13. Djahromi AK, Eltawil AM, Kurdahi FJ, Kanj R (Mar 2007) Cross layer error exploitation for aggressive voltage scaling. IEEE international symposium on quality electronic design, San Jose, CA, pp 192–197
14. Flautner K, Nam SK, Martin S, Blaauw D, Mudge T (2002) Drowsy caches: simple techniques for reducing leakage power. In: International symposium on computer architecture, Anchorage, AK, pp 148–157
15. George J, Marr B, Akgul BES, Palem KV (Oct 2006) Probabilistic arithmetic and energy efficient embedded signal processing. In: International conference on compilers, architecture, and synthesis for embedded systems, Seoul
16. Hesterberg TC (1988) Advances in importance sampling, Ph.D. Dissertation, Statistics Department, Stanford University
17. Horne S, Klein RK, Selcuk AA, Kepler NJ, Spence CA, Lee RT, Holst JC (Aug 1998) Memory device using a reduced word line voltage during read operations and a method of accessing such a memory device. US Patent 5796651
18. Jess JAG, Kalafala K, Naidu SR, Otten RHJ, Visweswariah C (2003) Statistical timing for parametric yield prediction of digital integrated circuits. In: Design automation conference, Anaheim, CA, pp 932–937
19. Kanj R, Joshi R, Nassif S (2006) Mixture importance sampling and its application to the analysis of SRAM designs in the presence of rare failure events. In: Design automation conference, San Francisco, CA, pp 69–72

20. Kawahara T, Sakata T, Itoh K, Kawajiri Y, Akiba T, Kitsukawa G, Aoki M (Jul 1993) A high-speed, small-area, threshold-voltage mismatch compensation sense amplifier for gigabit-scale DRAM arrays. IEEE J Solid State Circuits 28:816–823
21. Kurdahi F, Eltawil A, Djahromi AK, Makhzan M, Cheng S (Aug 2007) Error aware design.In: Euromicro conference on digital system design architectures, Lübeck, Germany, pp 8–15
22. Mojumder NN, Mukhopadhyay S, Kim JJ, Chuang CT, Roy K (Jan 2010) Self-Repairing SRAM using on-chip detection and compensation. IEEE Trans VLSI Syst 18(1):75–84
23. Mukhopadhyay S, Mahmoodi H, Roy K (Nov 2004) Statistical design and optimization of sram cell for yield enhancement. IEEE/ACM international conference on computer aided design, San Jose, CA, pp 10–13
24. Mukhopadhyay S, Mahmoodi H, Roy K (Dec 2005) Modeling of failure probability and statistical design of SRAM array for yield enhancement in nano-scaled CMOS. IEEE Trans Compu Aided Des Integ Circuits Sys 24(12):1859–1880
25. Mukhopadhyay S, Kim K, Mahmoodi H, Datta A, Park D, Roy K (Jun 2006) Self-repairing SRAM for reducing parametric failures in nanoscaled memory. In: Symposium on VLSI circuits, Honolulu, HI, pp 132–133
26. Mukhopadhyay S, Kim K, Mahmoodi H, Roy K (Jun 2007) Design of a process variation tolerant self-repairing SRAM for yield enhancement in nanoscaled CMOS. IEEE J Solid-State Circuits 42(6):1370–1382
27. Mukhopadhyay S, Mahmoodi H, Roy K (Jan 2008) Reduction of parametric failures in sub-100-nm SRAM array using body bias. IEEE Trans Comput Aided Des Integr Circ Sys 27(1):174–183
28. Mukhopadhyay S, Rao RM, Kim JJ, Chuang CT (2009) SRAM write-ability improvement with transient negative bit-line voltage. In: IEEE transactions very large scale integration (VLSI) systems
29. Mukhopadhyay S, Raychowdhury A, Mahmoodi H, Roy K (Dec 2005) Leakage current based stabilization scheme for robust sense-amplifier design for yield enhancement in nano-scale SRAM. In: IEEE Asian test symposium, Calcutta, pp 176–181
30. Nassif SR (2001) Modeling and analysis of manufacturing variations. In: Custom integrated circuit conference, San Diego, CA, pp 223–228
31. Ohbayashi S et el. (Apr 2007) A 65 nm SoC embedded 6T-SRAM design for manufacturing with read and write cell stabilizing circuits. IEEE J Solid State Circuits 42(4):820–829
32. Papoulis A (2002) Probability, random variables and stochastic process. MacGraw-Hill, New York, NY
33. Parke SA (1997) Optimization of DRAM sense amplifiers for the gigabit era. In: Midwest symposium on circuits and systems, Sacramento, CA, pp 209–212
34. Qin H, Cao Y, Markovic D, Vladimirescu A, Rabaey J (Mar 2004) SRAM leakage suppression by minimizing standby supply voltage. International symposium on quality electronic design, San Jose, CA, pp 55–60
35. Rabaey JM, Chandrakasan A, Nikolic B (2003), Digital integrated circuits. Prentice-Hall, ch. 12, Upper Saddle River, NJ, pp 623–718
36. Rao R, Devgan A, Blaauw D, Sylvester D (Jun 2004) Parametric yield estimation considering leakage variability. In: Design automation conference, San Diego, CA, pp 442–447
37. Roy K, Mukhopadhyay S, Mahmoodi H (Feb 2003) Leakage current mechanisms and leakage reduction techniques in deep-submicrometer CMOS circuits. Proc IEEE 91(2): 305–327
38. Sarpeshkar R, Wyatt JL, Lu NC, Gerber PD (Oct 1991) Mismatch sensitivity of a simultaneously latched CMOS sense amplifier. IEEE J Solid State Circuits 26:1413–1422
39. Schueller G, Pradlewarter H, Koutsourelakis PS (2003) A comparative study of reliability estimation procedures for high dimensions. In: ASCE engineering mechanics conference, Seattle, WA
40. Sub JW, Rho KM, Park CK, Koh YH (Jul 1996) Offset-trimming bit-line sensing scheme for gigabitscale DRAM's. IEEE J Solid State Circuits 31(7):1025–1028

41. Tang X, De V, Meindl JD (Dec 1997) Intrinsic MOSFET parameter fluctuations due to random dopant placement. IEEE Trans VLSI Syst 5:369–376
42. Taur Y, Ning TH (1998) Fundamentals of modern VLSI devices. Cambridge University Press, New York, NY
43. Visweswariah C (2003) Death, taxes and failing chips. In: Design automation conference, Anaheim, CA, pp 343–347
44. Wah BW, Chen Y-X (Nov 2000) Constrained genetic algorithms and their applications in nonlinear constrained optimization. In: IEEE international conference on tools with artificial intelligence, Vancouver, BC, pp 286–293
45. Wang Z, Bovik AC, Sheikh HR, Simoncelli EP (Apr 2004) Image quality assessment: from error visibility to structural similarity. IEEE Trans Image Process 13(4):600–612
46. Wicht B, Nirschl T, Schmitt D-Landsiedel (Jul 2004) Yield and speed optimization of a latch type voltage sense amplifier. IEEE J Solid State Circuits 39:1148–1158
47. Yamaoka M, Maeda N, Shimazaki Y, Osada1 K (Feb 2008) A 65 nm low-power high-density SRAM operable at 1.0 V under 3σ systematic variation using separate Vth monitoring and body bias for NMOS and PMOS. In: IEEE international solid state circuit conference, San Francisco, CA, pp 383–385
48. Yang S, Wolf W, Vijaykrishnan N (Jun 2005) Power and performance analysis of motion estimation based on hardware and software realizations. In: IEEE transactions on computers, pp 714–716
49. Yeung J, Mahmoodi H (Sep 2006) Robust Sense Amplifier Design Under Random Dopant Fluctuations In Nano-Scale CMOS technologies. In: IEEE international systems-on-chip conference, Austin, TX, pp 261–264
50. Yi K, Cheng SY, Kurdahi F, Eltawil A (2008) A Partial memory Protection scheme for higher effective yield of embedded memory for video Data. In: Asia-Pacific computer system architecture conference, Hsinchu, Taiwan, pp 273–278
51. Zhang K, Bhattacharya U, Chen Z, Hamzaoglu F, Murray D, Vallepalli N, Wang Y, Zheng B, Bohr M (Feb 2005) A 3-GHz 70 MB SRAM in 65 nm CMOS technology with integrated column-based dynamic power supply. In: International solid state circuits conference, San Francisco, CA, pp 474–611

Chapter 6
Digital Subthreshold for Ultra-Low Power Operation: Prospects and Challenges

Bipul C. Paul and Arijit Raychowdhury

Abstract This chapter provides a brief overview of the principles and challenges of ultra-low power circuit design in the subthreshold region of operation. Design principles at all levels of hierarchy, namely, devices, circuits, and architecture need to be evaluated for maximum power gains. Brief description of SRAM design techniques as well as alternative architectures for lower power has also been discussed.

6.1 Introduction

Computing with subthreshold leakage current has gained a wide interest in recent years to achieve ultra-low power consumption in portable computing devices [1–4]. In this mode of operation, the supply voltage is lowered below the threshold voltage of transistors so that all transistors are operated in the subthreshold region. Consequently, a significant reduction in power consumption can be achieved. However, a considerable decrease in performance will also occur due the exponential reduction in current in the subthreshold region. Therefore, subthreshold operation can only be used in applications requiring low to medium (ten to hundreds of megahertz) frequency of operation [5].

Both logic [2–9] and memory [10–12] circuits for subthreshold operation have been extensively studied with design consideration at various levels of abstraction [13–16]. Further, besides using conventional transistors operated in subthreshold region ($V_{\mathrm{DD}} < V_{\mathrm{TH}}$), optimizations of the transistor structures have also been investigated to achieve better results [17–19]. It is shown that halo and retrograde doping profiles can potentially be eliminated in planar devices for subthreshold operation (due to low V_{DD} and hence reduced short channel effects). Instead a high-low doping profile is suitable to achieve better subthreshold slope and lower junction capacitance [17]. Further, it has also been shown that unlike minimum possible oxide

B.C. Paul (✉)
Global Foundries, Fishkill, NY, USA
e-mail: bipul.paul@globalfoundries.com

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_6,

thickness (T_{ox}) offered by the technology, a thicker T_{ox} might be better for minimum energy operation depending on the technology [19].

In circuit and architecture levels, various logic styles [2–9], memory design [10–12], and architecture techniques [5, 20–23] have been explored. Besides conventional static CMOS logic, pseudo-NMOS [5], body biasing techniques such as DTMOS (dynamic threshold MOS) and source-coupled logics [3, 7–9] are shown to be suitable for subthreshold operation. Optimal transistor sizing is also a challenging issue in subthreshold design and have been addressed by different research groups [13, 14]. Various architecture techniques, both with clock (synchronous) and clockless (asynchronous) designs with pipeline and parallel architectures have also been studied to optimize the energy and performance of subthreshold operation [5, 20–23]. In this chapter we will provide an overview of an optimal system design in the subthreshold regime by careful co-optimization of devices, circuits, and architecture. It should be noted that the exponential nature of current in the subthreshold region makes it more prone to process and environmental variations. Hence, variability-aware design techniques are of critical necessity.

Increased variations render increased challenges in the design of large arrays. SRAMs are particularly susceptible to these variations because of the reduced transistor on-to-off ratio due to small supply voltage and dramatic increases of within die and die-to-die variations [16]. SRAM design for subthreshold operation requires careful consideration to achieve acceptable circuit reliability and performance even under variations. In this chapter, we will discuss various device and circuit/architecture design techniques for robust subthreshold operation for both logic and memory.

6.2 Fundamentals of Subthreshold Operation

The transistor current, which varies exponentially with gate input voltage, V_{GS}, is used as the operating current in subthreshold operation (Fig. 6.1) [17]. Since the transistor is operated in the subthreshold region, both ON($V_{\mathrm{GS}} = V_{\mathrm{DD}}$; $V_{\mathrm{DD}} < V_{\mathrm{TH}}$) and OFF ($V_{\mathrm{GS}} = 0$) currents are drain–source diffusion currents as given by [24],

$$I_{\mathrm{ds}} = \frac{W_{\mathrm{eff}}}{L_{\mathrm{eff}}} \mu_{\mathrm{eff}} \sqrt{\frac{q\varepsilon_{\mathrm{Si}} N_{\mathrm{eff,ch}}}{2\Phi_{\mathrm{s}}}} \beta^2 \exp\left(\frac{V_{\mathrm{GS}} - V_{\mathrm{TH}}}{\eta\beta}\right) \left[1 - \exp\left(\frac{-V_{\mathrm{DS}}}{\beta}\right)\right] \tag{6.1}$$

where $N_{\mathrm{eff,ch}}$ is the effective channel doping [17], Φ_{s} is the surface potential, and β is the thermal voltage given by kT/q. μ_{eff} is the effective carrier mobility in the channel, η corresponds to the subthreshold swing, and $\varepsilon_{\mathrm{Si}}$ is the permittivity of silicon. V_{TH} is the threshold voltage of the transistor given by [24],

$$V_{\mathrm{TH}} = V_{fb} + \Phi_s + \gamma\sqrt{\Phi_{so} - V_{\mathrm{bs}}}\left(1 - \lambda \frac{X_{\mathrm{d}}}{L_{\mathrm{eff}}}\right) + \Delta V_{\mathrm{NWE}} \tag{6.2}$$

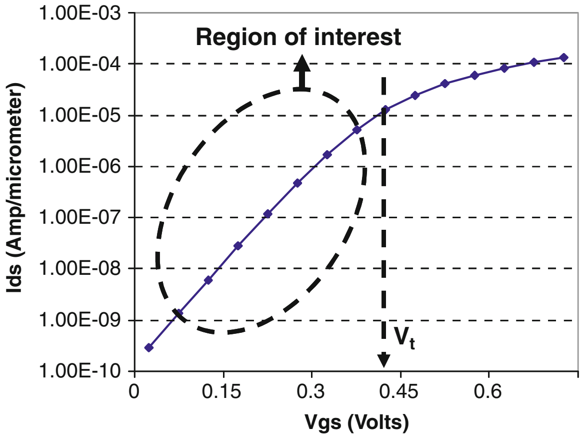


Fig. 6.1 I_{DS} vs. V_{GS} of a transistor in the subthreshold region of operation [17]

where γ is the body factor, X_d is the short channel correction factor and ΔV_{NWE} is the narrow-width correction factor given in [25]. Because of the exponential $I-V$ characteristics, we get a very high transconductance (g_m) of the device, which results in near ideal voltage transfer characteristics (VTC) of the logic gates and enables us to design more complex gates. This is because, I_{ds} in the strong inversion region saturates when $V_{DS} > V_{GS} - V_{TH}$, while it saturates when $V_{DS} > \sim 3kT/q(\sim$ 78 mV at 300 K) in the subthreshold region. In subthreshold operation the effective device capacitance, which is a combination of gate oxide capacitance, depletion capacitance, and various parasitic capacitances (overlap capacitances, fringing capacitances, etc. (see Fig. 6.2)), is also lower than the gate oxide capacitance. The effective capacitance, C_G, of a transistor can be represented as

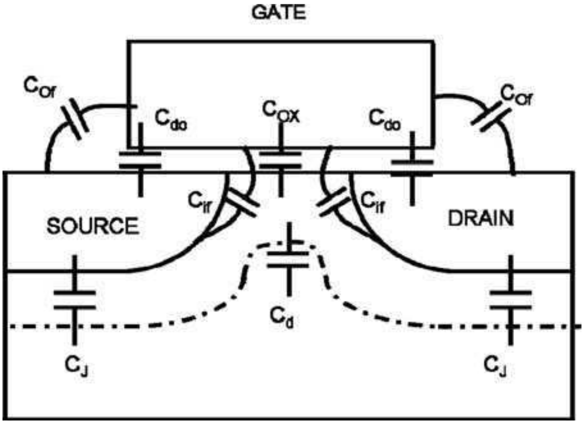


Fig. 6.2 Schematic of a MOSFET with all its capacitance components (intrinsic and parasitic) [17]

$$C_{\mathrm{G}} = \mathrm{series}(C_{\mathrm{ox}}, C_{\mathrm{d}})||C_{\mathrm{if}}||C_{\mathrm{of}}||C_{\mathrm{do}} \tag{6.3}$$

where C_{ox} is the oxide capacitance, C_{d} the depletion capacitance, and C_{do}, C_{if}, and C_{of} are the parasitic overlap and fringe capacitances, respectively. Note that in subthreshold operation both inner (C_{if}) and outer (C_{of}) fringe capacitances contribute to the overall device capacitance (C_{G}). In contrast, the effective device capacitance in strong inversion operation is dominated by the gate oxide capacitance. Due to the smaller device capacitance and lower supply voltage digital subthreshold circuits consume less power at a target frequency of operation than their strong inversion counterpart. For example, a subthreshold circuit ($V_{\mathrm{DD}} = 0.5\,\mathrm{V}$) operated at 10 kHz consumes about four orders of magnitude less power than its strong inversion counterpart ($V_{\mathrm{DD}} = 3\,\mathrm{V}$).

The above characteristics emphasize the need of different design approaches compared to superthreshold (standard strong inversion) operation both at the device and circuit level. Consequently, various device design and circuit and architecture techniques have been explored to optimize the performance of subthreshold operation, i.e., to minimize energy consumption while maximizing the speed. In the following sections we will discuss the effective device and circuit/architecture design techniques to achieve this goal.

Further, although there are a number of favorable changes, such as increased transconductance gain and near-ideal static noise margin, the sensitivity of subthreshold circuit to power supply, temperature, and process variations, however, increases due to the exponential nature of $I - V$ characteristics. The increase in this sensitivity may cause subthreshold circuits to fail to function properly. Subthreshold designs, hence, require special care to mitigate this effect and certain techniques to this effect would be discussed in this chapter.

6.3 Device Optimization

Typically, in subthreshold operation, the conventional MOSFET transistors are operated in the subthreshold region. In order to ensure that the entire circuit is indeed operating in the subthreshold region, a power supply less than the threshold voltages of the transistors is used to power the circuit. Consequently, due to the absence of conducting inversion channels, transistors behave differently as compared to when they are operated in a normal strong inversion region. Therefore, devices should be designed differently for subthreshold operation to achieve lower energy and higher performance. In this section we will discuss the optimization of two major device design parameters: (1) doping profile and (2) gate oxide thickness.

6.3.1 Doping Profile Optimization

Scaled device technologies demand non-uniform doping profiles to provide good control on the electrical characteristics of the device. It is an established fact that for scaled superthreshold transistors it is essential to have halo and retrograde doping to suppress the short channel effects. The main functions of halo doping and retrograde wells are to reduce drain induced barrier lowering (DIBL), prevent body punch-through, and control the threshold voltage of the device independent of its subthreshold slope. However, in subthreshold operation, it is worthwhile to note that the overall supply bias is small (in the order of 150 mV $\sim$ 300 mV). Consequently, the effects of DIBL and body punch-through are extremely low. Hence, it can be qualitatively argued that the halo and retrograde doping are not essential for subthreshold device design. The absence of the halo and retrograde doping has the following implications:

- *Arguably a simplified process technology in terms of process steps and cost.*
- *A significant reduction of the junction capacitances resulting in faster operation and lower power.*

It should, however, be noted that the doping profile in these optimized devices should have a high-to-low profile. It is necessary to have a low doping level in the bulk of the device to

- *Reduce the capacitance of the bottom junction.*
- *Reduce substrate noise effects and parasitic latch-up problems.*

The detailed description of the optimization process and the implications on the device design can be found in [17]. It has been shown that the optimized subthreshold device provides smaller delay owing to its higher on-current and reduced junction capacitance. Simulations show that the delay decreases by 44% when the optimized devices were used. Further, the reduced junction capacitances result in reduced dynamic (or switching) power. Figure 6.3 shows the power delay product (PDP), which is a measure of the switching energy, of the inverter circuits built with (i) *standard devices* and (ii) *subthreshold devices* for two different values of V_{DD} (the devices are in subthreshold for both the V_{DD} values). The optimized subthreshold devices provide a significant improvement in PDP clearly demonstrating the effectiveness of device design and optimization specifically for subthreshold operation.

6.3.2 Oxide Thickness Optimization

Similar to the doping profile optimization, the oxide thickness for subthreshold devices need to be designed to provide the minimum total power. In conventional technologies, a minimum possible oxide thickness (T_{ox}) provided by the technology is targeted for better electrostatics and improved subthreshold slope. This, however, comes at the expense of increased gate capacitance. Details of such optimizations have been presented in [19] and here we will present the key result. It is observed

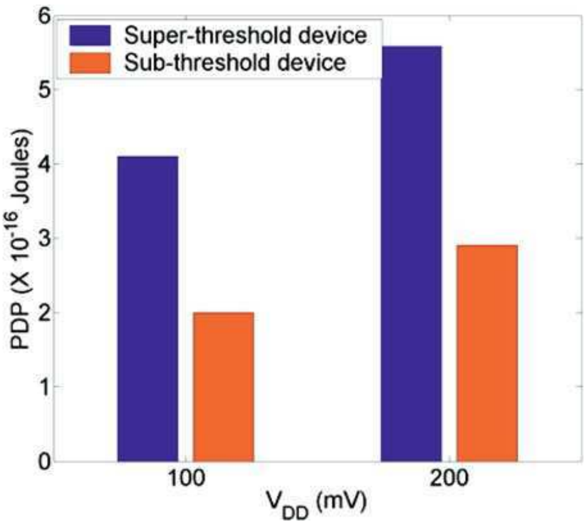


Fig. 6.3 Variation of PDP with the V_{DD} for an inverter driving an identical inverter (for the super-threshold and the optimized subthreshold device both operated in subthreshold region) [17]

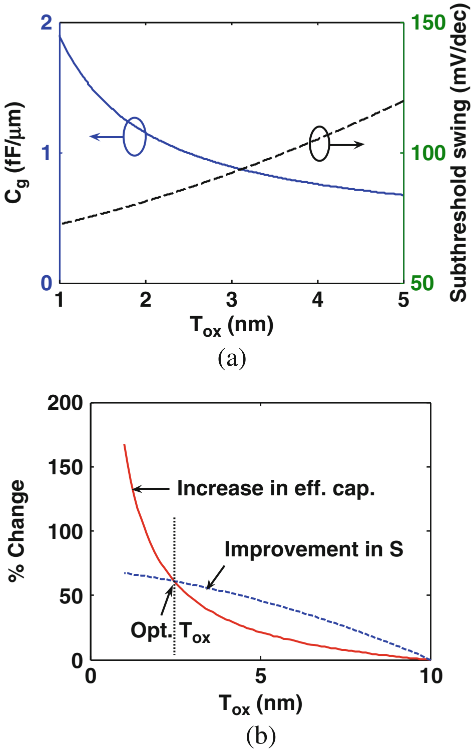
that the required V_{DD} for constant I_{ON} reduces with T_{ox} as expected. However, the dynamic power (E_{dyn}) does not monotonically reduce with oxide thickness and the minimum occurs around a T_{ox}, which is slightly larger than the minimum T_{ox} (1.2 nm) offered by the technology. Further, the optimum T_{ox} (corresponding to minimum energy) is approximately the oxide thickness, where the increased power dissipation due to an increase in C_{G} is not amortized by the improvement in subthreshold swing (Fig. 6.4). It was also noted that optimum T_{ox} has a weak dependency on the fanout, and the variation in minimum E_{dyn} is less than 2%.

The above discussion demonstrates the need of device design specifically for subthreshold operation. Similar efforts have been carried out in non-planar devices and the interested readers are referred to [19] for further discussions. The performance of subthreshold operation can further be improved by utilizing innovative circuits and architectures. We will discuss various circuit and architectural techniques suitable for subthreshold operation in the following sections.

6.4 Circuit and Architecture Optimization

In this section, we will discuss circuit and architecture level design techniques for subthreshold operation. A few logic styles, memory and architecture techniques suitable for subthreshold operation will be described here. Interested readers can refer to a variety of other techniques provided in the reference section.

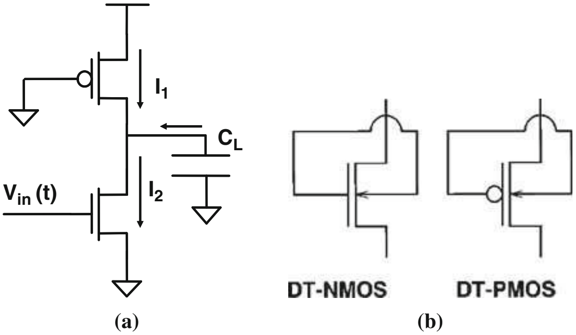
Fig. 6.4 (**a**) Change in effective gate capacitance (C_g) and sub-threshold swing (S) with T_{ox}. The analysis was done considering 90 nm technology node with 50 nm effective channel length; (**b**) Percentage variation in C_g



6.4.1 Subthreshold Logic Styles

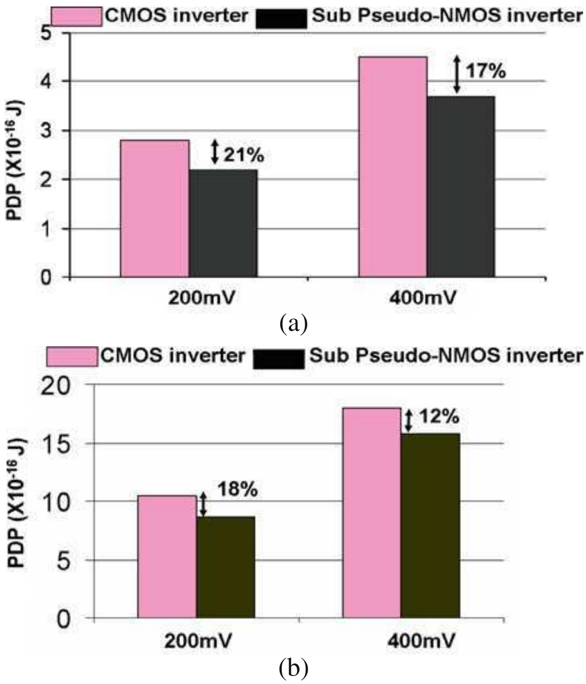
Although different logic styles, such as pseudo-NMOS, dynamic threshold (DTMOS), source coupled logics, etc., have been explored for subthreshold operation besides the conventional static CMOS logics, we will discuss pseudo-NMOS and DTMOS logics here and compared them (Fig. 6.5a, b) with the static CMOS logic circuits. Readers are, however, encouraged to refer to other logic styles provided in the reference section in this chapter.

Fig. 6.5 (**a**) Pseudo-NMOS logic and (**b**) DT-MOS Logic Styles



6.4.1.1 Sub-Pseudo-NMOS Logic

Pseudo-NMOS logic (Fig. 6.5a) is faster than static CMOS due to smaller load capacitance and shorter interconnects. However, in order to utilize pseudo-nMOS logic, the drawbacks of ratioed logic such as large static current consumption and degradation in static noise margin should be carefully taken into account. Pseudo-NMOS logic in the subthreshold region (sub-pseudo-NMOS) inherits the advantages it has in the strong inversion such as higher performance and smaller area. In addition to this, the drawbacks of ratioed logic are relieved in the subthreshold region [26]. This is mainly because in the subthreshold region, the drain current saturates and becomes independent of V_{DS} for $V_{\mathrm{DS}} > \sim 3kT/q$ (78 mV at 300 K). Note that a transistor in strong inversion region (super-threshold) only enters the saturation region when $V_{\mathrm{DS}} > V_{\mathrm{GS}} - V_{\mathrm{TH}}$, which gives a much narrower saturation region, and thus, an undesirable voltage transfer characteristic (VTC). However, the output voltage of pseudo-NMOS in subthreshold region swings for almost rail-to-rail, and thus provides a high noise margin. When both the super-threshold and subthreshold pseudo-NMOS gates are sized for identical low-to-high and high-to-low delays, the subthreshold gate provides a better noise margin and a sharper VTC. Sub-pseudo-NMOS is also more efficient than sub-CMOS in terms of PDP. The PDP of a pseudo-NMOS inverter (driving an identical inverter as well as a fan-out of four) is compared to a CMOS inverter in Fig. 6.6 [5]. In the subthreshold region, pseudo-NMOS provides approximately 20% improvement in PDP compared to CMOS. The



(a)

(b)

Fig. 6.6 The PDP of an inverter in CMOS and Sub Pseudo-NMOS logic styles made with the optimized devices driving (**a**) an identical inverter, (**b**) four identical inverters [5]

reason behind the lower PDP in sub-pseudo-NMOS is the smaller delay. Further, in the subthreshold region the static short circuit current is also a weak inversion current, which is relatively much less significant than in super-threshold.

It should, however, be noted that the pseudo-NMOS logic style suffers from a static leakage, when the pull-down network in ON. In subthreshold domain, however, this ON current is in the orders of magnitude smaller than super-threshold. Hence, in cases where the switching activity is extremely low, the total power consumption in the pseudo-NMOS logic gate can be potentially higher than CMOS.

6.4.1.2 Sub-DTMOS Logic

Subthreshold dynamic threshold MOS (sub-DTMOS) logic uses transistors whose gates are tied to their substrates (Fig. 6.5b). As the substrate voltage in sub-DTMOS logic changes with the gate input voltage, the threshold voltage is dynamically changed. In the off-state, i.e., $V_{\mathrm{in}} = 0$ ($V_{\mathrm{in}} = V_{\mathrm{DD}}$) for NMOS (PMOS), the characteristics of DTMOS transistor is exactly the same as regular MOS transistor. Both have the same properties, such as the same off-current, subthreshold slope, and threshold voltage. In the on-state, however, the substrate–source voltage (V_{bs}) is forward biased and thus reduces the threshold voltage of DTMOS transistor. The reduced threshold voltage is due to the reduction in body charge [27]. The reduction in body charge leads to another advantage, namely higher carrier mobility because the reduced body charge causes a lower effective normal field. The reduced threshold voltage, lower normal effective electric field, and higher mobility results in higher on-current drive than that of a regular MOS transistor. Furthermore, the subthreshold slope of DTMOS improves and approaches the ideal 60 mV/decade which makes it more efficient in subthreshold logic circuits to obtain higher gain. Another significant advantage of the sub-DTMOS logic is that it does not require any additional limiter transistors, which further reduces the design complexity. In contrast, in the normal strong inversion region, the limiter transistors are necessary to prevent forward-biasing the parasitic PN junction diode while allowing a much higher power supply to be used in the circuit. The delay and power-delay-product (PDP) of an example sub-DTMOS ring oscillator is compared with its sub-CMOS counterpart (Fig. 6.7). The higher on-current of sub-DTMOS logic causes it to have higher power consumption, but can switch much faster than regular sub-CMOS logic. Although DTMOS gate capacitance is larger than that of standard MOS gate capacitance, gate capacitance is only a portion of the total capacitance and the increase in current drive of DTMOS far outweighs the increase in gate capacitance. Thus, a DTMOS gate switches faster than a regular MOSFET. This can be further understood from the PDP of both circuits. Since the PDP of the both circuits are comparable, the sub-DTMOS logic can be operated at much higher frequency while maintaining the same switching energy.

Similar to sub-pseudo-NMOS logic, DTMOS transistors can also be used to implement DT-pseudo-NMOS logic to be operated in subthreshold operation (sub-DT-pseudo-NMOS logic). Two different types of sub-DT-pseudo-NMOS circuit

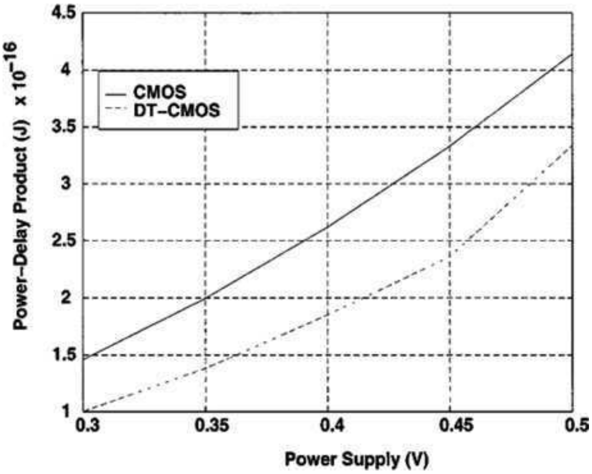


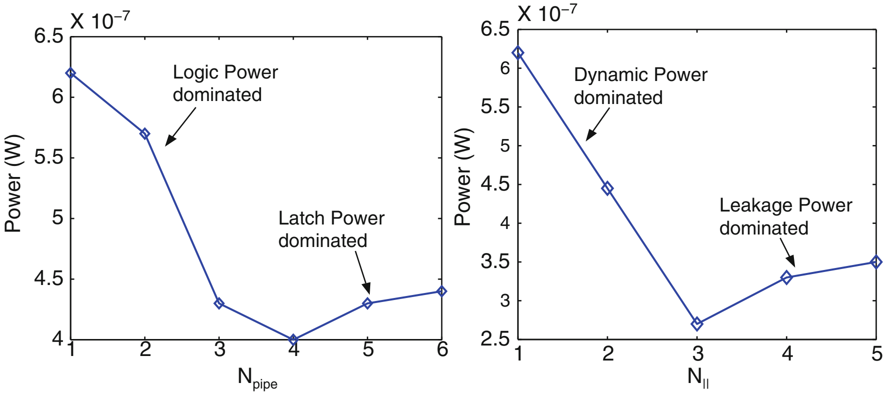
Fig. 6.7 PDP of sub-CMOS and sub-DTMOS logic ring oscillator circuits at 0.35 µm technology [3]

implementations were studied in [3]. All NMOS transistors in the pull-down network (PDN) are replaced with DT-NMOS transistors. Sub-DT-pseudo-NMOS logic implementations have better delay and lower PDP than the regular sub-pseudo-NMOS logic circuit. However, a caveat for using pseudo-NMOS is the increased leakage current, which makes it power inefficient for low switching activities.

6.4.2 Parallelizing and Pipelining

Since the primary objective of system design in subthreshold operation is low power, the focus is on searching the design space for minimizing total power while maintaining a target throughput (or maximizing throughput at a constant power). Further, it may not be possible to meet a target throughput by device and circuit optimization only. Hence, it is necessary to exploit architectural level techniques to obtain higher throughput at lower power.

Parallelism and pipelining are two effective ways to improve system performance during architectural synthesis [5]. Typically, they represent a trade-off between area/power and system throughput. Parallelism is associated with incorporating more hardware resources to increase throughput. However, more parallel elements result in increased leakage power and die-area. On the other hand, the idea of a pipelined design is associated with increasing operating frequency (and hence throughput), by breaking the critical signal propagation path of a circuit into multiple smaller paths (with latch insertion at appropriate points in the pipeline). However, the increase in number of pipeline stages corresponds to an increase in overhead due to latches (in terms of die-area and clock power) and the latency of operation (Fig. 6.8a).



(a) (b)

Fig. 6.8 For a five-tap FIR filter at 10 MIPS (**a**) variation of power with number of pipeline stages (one parallel unit), (**b**) variation of power with number of parallel units (one pipeline stage). $V_{\rm DD}$ has been adjusted in each case to achieve iso- throughput [5]

The reduction in supply voltage is a popular choice to reduce power dissipation since it renders quadratic savings in dynamic power. The performance loss due to supply voltage scaling can be compensated by architectural means using more parallel resources and/or increasing the number of pipeline stages as mentioned above. However, in subthreshold region the dynamic power constitutes a smaller percentage of the total power than in super-threshold. This can be understood from the following:

- *gate input capacitance is lower than its super-threshold counterpart.*
- *operating voltage, V_{DD}, as well as the operating frequency, f, are lower than in super-threshold.*

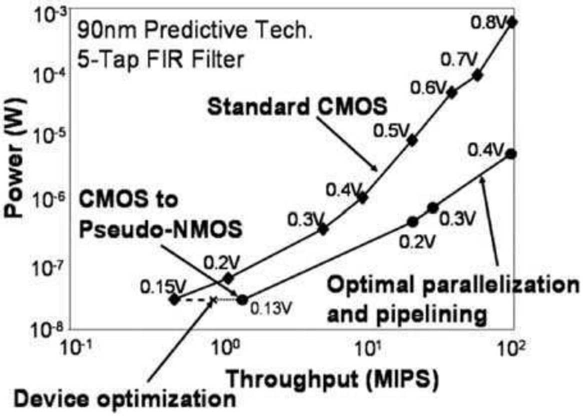
Even for the same frequency of operation, a subthreshold design has lower dynamic power than the super-threshold design. Consequently, even in the active mode of the circuit, the leakage power is a significant portion of the total power.

To maintain a constant throughput, we can lower $V_{\rm DD}$ if we incorporate more parallel processing units. This, of course, decreases the dynamic power of the system. However, the leakage power increases steadily (because of more number of processing units) and very soon becomes the dominant portion of the total power. In subthreshold operation this trade-off becomes apparent for a relatively lesser number of parallel units. Hence, contrary to superthreshold designs that parallelization can ideally reduce power for constant throughput (by aggressively scaling $V_{\rm DD}$), in subthreshold it is necessary to judiciously choose the number of parallel units such that the total power (dynamic + leakage) is minimized.

Pipelining too offers a power-frequency trade-off in terms of the contributions from the latches. The latches contribute significantly not only to the dynamic but also the leakage power. Hence, the number of pipeline stages need to be chosen so that the total power (combinational logic + latches) is minimized. Thus, there is a need for identifying the global optimal in terms of number of parallel units and pipeline stages for any system design (Fig. 6.8b).

Figure 6.9 illustrates the device/circuit/architecture co-optimization of subthreshold circuits as applied to an FIR filter. For more details on the choice of circuits and architectural techniques, the readers are being referred to [5].

Fig. 6.9 Throughput vs. Power of a five-tap FIR filter. Note how device optimization, choice of circuit style and optimal parallelization/pipelining can significantly improve throughput at iso-power



6.4.3 Event-Driven CPU Architectures

One of the techniques to reduce the power consumption of a system is to activate a block only when it is required to function. The enable/disable signal for a block is generated in the control unit based on the outputs of the previous stages. Such an event-driven approach leads to considerable power savings since the unnecessary switching in the modules of the system is prevented. The Phoenix processor [28] uses such a technique.

The Pheonix processor is modular in nature with a CPU, memory units, and a power management unit (PMU). The operation of the system begins in the sleep mode. Only the PMU remains active in this mode. For the rest of the modules, the transistors are gated using footer transistors. To reduce leakage in the PMU, stacked high V_{TH} devices are used. After a programmable sleep time, an interrupt is issued which activates the CPU to retrieve and process the sensor data. After some time, the system goes back into the sleep mode. Such an event-driven architectural technique helps to reduce the overall power consumption of this processor.

In addition to this, a couple of techniques have been used to reduce the footprint of the memory units: (1) a reduced instruction set is used in order to minimize the instruction memory (IMEM) footprint. (2) Data compression techniques like Huffman encoding are used to lower the data memory (DMEM) footprint and to achieve higher memory capacity. Since a large fraction of sleep mode power is consumed in these units, employing such techniques leads to large power savings.

6.5 SRAM Design for Subthreshold or Near Subthreshold Operation

The increasing need for functionality and the aggressive scaling of transistor dimensions has led to larger caches and register files in microprocessors. Memory design is, in general, more stringent as compared to logic design because of the following reasons: (1) standby leakage power becomes more important since most part of the memory array does not switch, (2) stability of the cells (defined in terms of static noise margin, SNM) needs to be taken into consideration, which involves trade-off with speed and power, (3) read and write operations have conflicting requirements, and (4) process variations, both die-to-die (D2D) and within-die (WID), play a more significant role. These factors become further exacerbated in the subthreshold region, because of reduced supply voltage and high sensitivity of circuits to process variations. Simple bitcell structures like 6-T cells, shown in Fig. 6.10, do not meet the requirements for stability, readability, and write-ability at subthreshold regime. Both read and write assist techniques are widely used to enable low voltage operation of SRAM bitcells. Further, more complex bitcells that ensure correct functionality of the memory at low supply voltages and in the presence of process variations, is under active research.

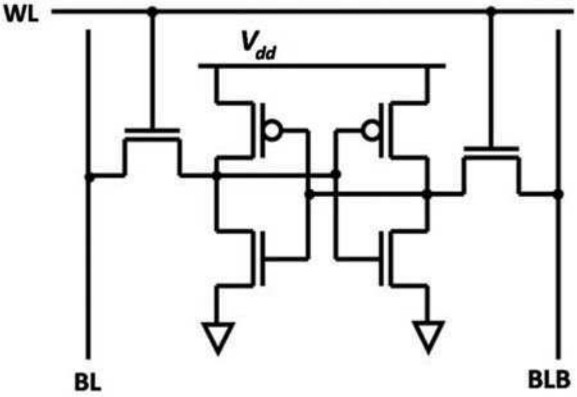


Fig. 6.10 Schematic diagram of a standard 6T SRAM bitcell

The designers' choice for low level, high density caches is the 6-T SRAM bitcells due to (i) low area and high array efficiency and (ii) lower standby power (see Fig. 6.10 for the schematic of 6-T SRAM cell). However, in the subthreshold region: (1) cell stability is heavily compromised because read and write ports are coupled. This problem aggravates in the subthreshold region because of low supply voltage and hence less noise margins. (2) Process variations (PV), which have larger effect in the subthreshold region, may degrade write-ability, stability, and readability. (3) Ratio of I_{on} to I_{off} is low which affects readability due to bitline leakage current [29]. Hence, it becomes important to use circuit techniques at the array or the bitcell level that mitigate these problems at the cost of extra area.

6.5.1 Alternative Bitcell Architectures: Non-differential Bitcells

In [30], an 8-T subthreshold bitcell is used to isolate the read and write operations. This mitigates the problem of cell stability during read. Figure 6.11 shows the decoupled read and write ports. To take care of write-ability degradation due to PV, PMOS of the cross-coupled inverter is weakened by reducing the supply voltage during the write operation. This is achieved by using a periphery supply driver shown. The pull-down devices of the supply driver tend to pull VV_{dd} (virtual V_{DD}) to zero. However, VV_{dd} does not go all the way to "0" since bitcells of the accessed row pull it up. This can be understood as follows. The node in the bitcell which stores "0" turns on PMOS (say, M4) of the other inverter. Since the drain voltage of M4 is "1", VV_{dd} is pulled up through all the accessed half-bitcells. Relative strengths of pull-up action of bitcells and pull-down action of supply driver determine VV_{dd}. Thus, supply voltage during write operation can be reduced to the desired value. Write-ability depends on the relative strengths of the pull-up PMOS of the cross-coupled inverter and write access transistor. Hence weakening PMOS in this manner improves write-ability, even for the worst case corner. Pull-down devices in the supply driver are large and hence, less susceptible to PV. The effect of PV on the pull-up mechanism of VV_{DD} by half-bitcells averages out. Therefore, the effect of PV on write-ability is reduced. Finally, since there is some leakage in the supply driver, two NMOS devices can be placed in series to reduce leakage by making use of the stacking effect [31].

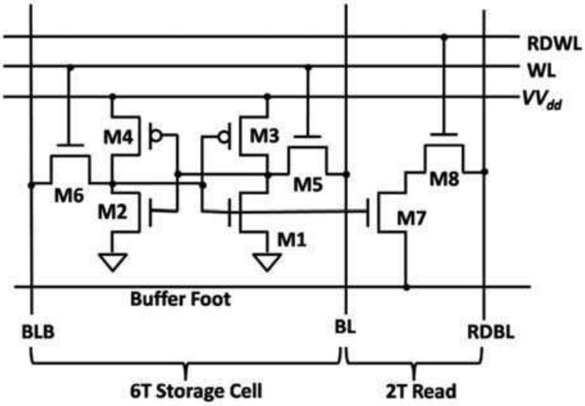
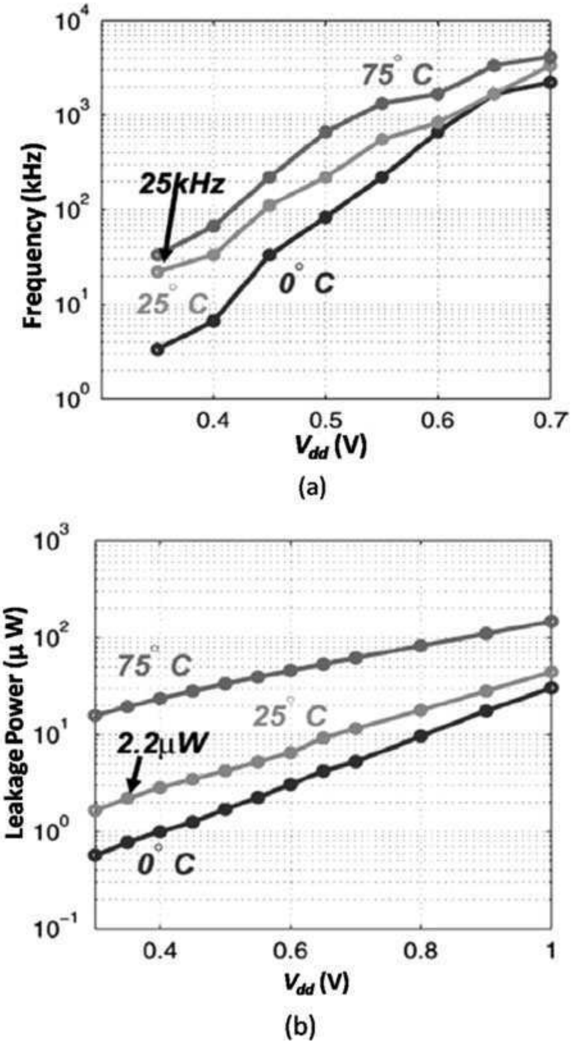


Fig. 6.11 Schematic diagram of an 8T SRAM bitcell featuring decoupled read and write ports

Fig. 6.12 Measured Frequency and Leakage power as a function of V_{DD} [30, 32]



Finally, to address the issue of bitline leakage on readability, sources of all the unaccessed read buffers are pulled up to V_{DD}. This ensures that the voltage drop across them is zero, which implies zero leakage current (Fig. 6.12)

However, the proposed technique requires the size of the NMOS of the buffer to be extremely large, since it has to sink the read current of the entire accessed row. To get around this problem, charge pump circuit is used, the details of which can be found in [30]. Further, to counter the effects of PV in sense amplifiers, a concept of sense amplifier redundancy is used, in which read bit line of each cell is connected to *N* sense amplifiers so that the probability of failure decreases by a power of *N*. A

256 KB memory with the proposed SRAM bitcell structure and peripheral circuits was fabricated in 65-nm technology. It can be seen that, for the supply voltage of 350 mV, frequency of the order of a 25 kHz was achieved with the leakage power of 2.2 μW at room temperature.

In [29], a write-back mechanism is proposed to improve stability of SRAM cells. During the write operation, the unselected columns suffer from stability degradation. The idea is to read the data during write operation. To achieve this, a writeback enabled signal is asserted after sufficient delay for the unselected columns, so that the value read is written at the end of the operation. This leads to stability improvements of the unselected columns.

The SRAM cell shown in Fig. 6.13 is used for the Pheonix processor [28]. It can be seen that read and write operations are decoupled in this cell. Since power optimization of the system is of utmost importance, high V_{TH} (HVT) transistors are used to reduce leakage. In order to complete the read operation in a single cycle, medium V_{TH} (MVT) transistors are used in the read buffer. Further, write operations are made asynchronous, since the use of HVT transistors increases the write time.

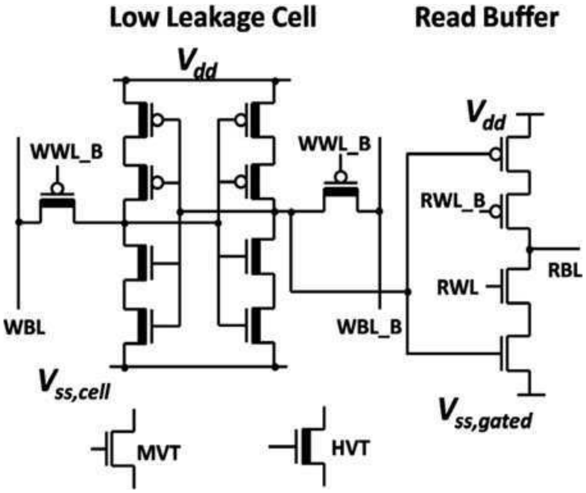


Fig. 6.13 A mix of HVT and MVT transistors in the SRAM bitcell achieves lower power in the Phoenix processor [28]

6.5.2 *Alternative Bitcell Architectures: Differential Bitcells*

In [33], an SRAM bitcell, based on the Schmitt trigger (ST) action, is proposed. The bitcell has better sensitivity to process variations, and therefore is a promising candidate for subthreshold memories. Figure 6.14a shows the schematic of the proposed structure and hereafter is referred to as the ST-1 cell. Suppose, V_{L} and V_{R} are storing logic "0" and "1" respectively. During the read operation, BL and BR are pre-charged to V_{DD}, WL turns on and V_{L} rises to some voltage V_{read}. Read failure occurs if V_{read} is greater than logic threshold voltage (V_{M}) of the inverter on the right

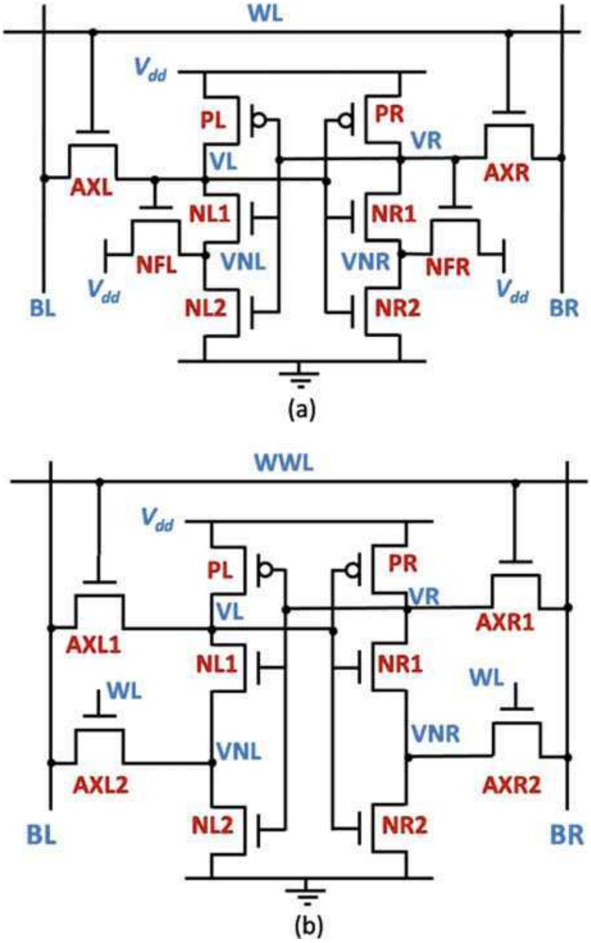


Fig. 6.14 Schmitt trigger based SRAM cells show better voltage scalability [33, 34]

side. Feedback transistor NFR charges V_{NR} to a voltage close to V_{DD}, making V_{gs} of NR1 negative. This increases V_{M} of the right hand side inverter, thus increasing read stability. During the write operation, BR is discharged to ground and BL is charged to V_{DD}. The node voltage at V_{R} drops down to some value determined by the relative strengths of PR and AXR. On the left hand side, V_{M} of the inverter increases due to the series NMOS in the pull-down network, making it easy to flip the voltage at V_{L}. Thus ST-1 cell has improved write-ability and read stability. In addition, feedback and pull-down transistors track each other across process corners, thus decreasing sensitivity to process variations.

Figure 6.14b shows another cell based on Schmitt trigger action proposed in [34] (hereafter referred to as ST-2 cell). This cell has improved read stability and write-ability compared to 6-T cells. Again assume V_{L} and V_{R} are storing logic “0” and

"1" respectively. During the read operation, WL is ON while WWL is OFF. V_{NL} rises to some voltage V_{read}, which is less than that for ST-1 cell. This is because there is one pull-down transistor (NL2) in the path of read current as opposed to two series transistors (NL1 and NL2) in case of ST-1 cell. Moreover, the gate of AXR2, which raises V_{M} of the right hand side inverter, is controlled by WL rather than the storage node, as in ST-1 cell. This makes the feedback action stronger. These factors contribute in increasing the read stability of ST-2 cell. During the write operation, both WL and WWL are turned on. In ST-2 cell, as V_{L} makes 0–1 transition, NR1 provides a path for the write current (through NR1 and AXR2) in addition to the usual path through AXR1. This results in improved write-ability compared to the standard 6T cell.

Because of improved read and write stabilities, ST-1 and ST-2 cells can be operated at a lower voltage, thus leading to reduction in dynamic and leakage power. This also lowers the minimum supply voltage for correct functionality of these cells, increasing the dynamic range of V_{DD}. It may be also noted that for ST-1 cell, there is no change in the bitline capacitance and the cell can be used as a drop-in replacement for 6T cell.

Figure 6.15 shows the characteristics of the Schmitt Trigger. Butterfly-curves in Fig. 6.15a, b illustrate that ST-1 cells exhibit superior read and hold stability compared to 6-T cells. Figure 6.15c, d show read and hold SNM and write margin of ST-1 and 6-T cells for different supply voltages. It can be seen that ST-1 cells have better read and hold stability as well as improved write stability. Similar results for the ST-2 cell have also been reported in [34].

Yet another structure (see Fig. 6.16) has been proposed in [35], which isolates the read and write operation without having a separate bitline for read. During the read operation, only WL is asserted, which does not affect the state of nodes Q and QB. During the write operation, both WL and WWL are asserted. Because of isolation of state nodes during read, read SNM is as high as the hold SNM. It can be observed that since the two access transistors are in series, write-ability suffers, but at the same time, there is a reduction in the bitline leakage current, which improves readability. In order to mitigate the effect of series access transistors on write-ability, word lines are boosted.

The discussion in the last two subsections is not exhaustive but presents a mere glance of the vast plethora of research activities in designing low voltage SRAM arrays. Apart from the alternative bitcell architectures, array level techniques to facilitate read and/or write, as well as the use of embedded PVT sensors have also become common.

6.6 Variability-Aware Design Techniques

It has already been discussed that process variations (PV) plays a significant role on the operation of the subthreshold circuits. The reason for this is the exponential dependence of current on the threshold voltage, which is one of the most susceptible parameters to process variations. Some of the ways in which the threshold

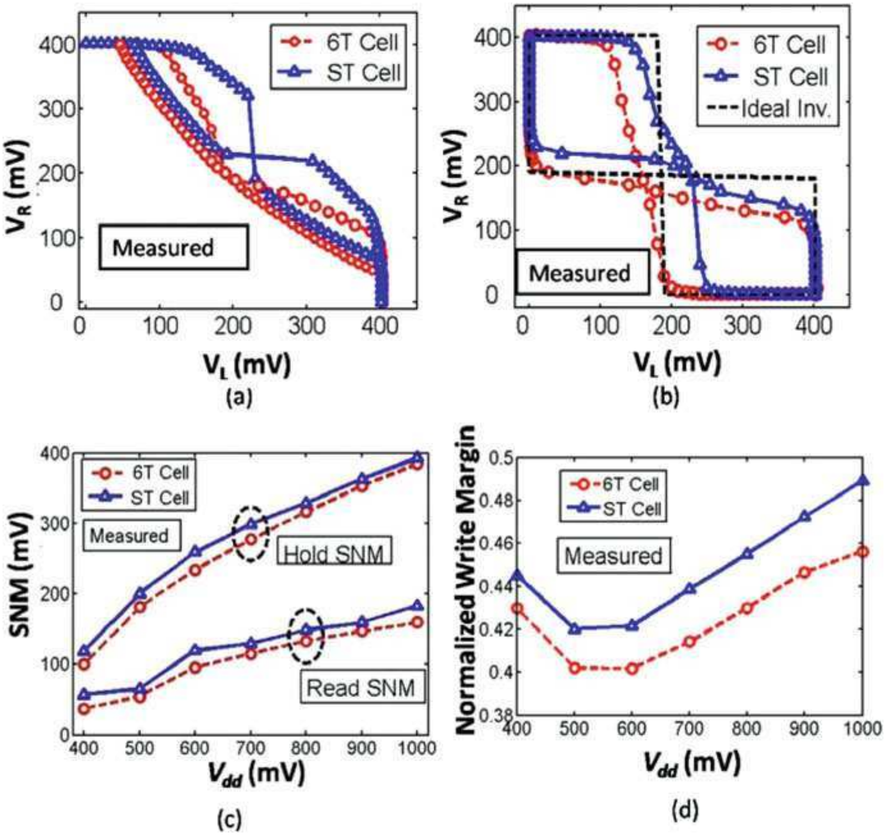


Fig. 6.15 Measured data from the schmitt trigger SRAM cell illustrates better voltage scalability [33, 34]

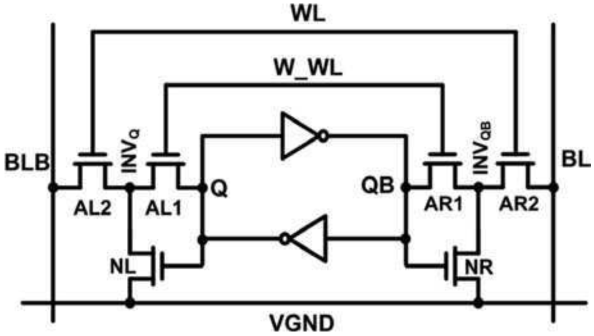
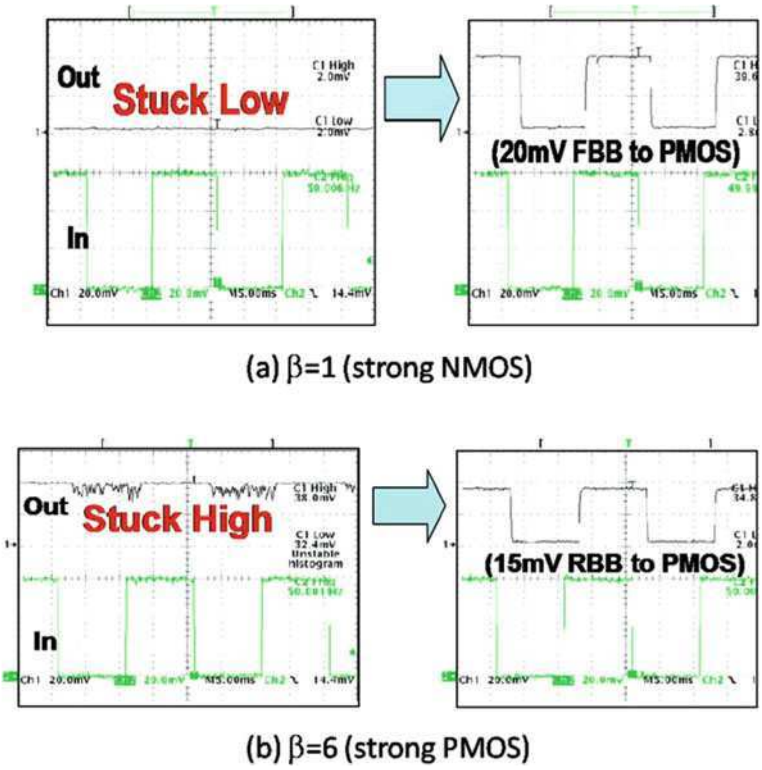


Fig. 6.16 The differential cell proposed in [35] features decoupled read and write and high read SNM

voltage may be affected by PV are (1) random dopant fluctuation, (2) variation in the oxide thickness, and (3) variation in channel length for highly scales devices. All these variations are commonly observed and hence, it should be expected that subthreshold variation will play a significant role in the design of circuits.

The impact of PV is most severe when the functionality of the system is affected. When circuits are designed at very low voltages in the subthreshold region, a simple circuit like a chain of inverters fails to give the expected output under process variations [36]. The reason can be understood as follows. The β ratio (i.e., the ratio of the PMOS strength to NMOS strength) for a subthreshold inverter is different from the conventional super-threshold one. This is because Idn–Idp ratio for equal width of NMOS and PMOS is not only affected by the difference in electron and hole mobilities, but also by the difference in V_{TH} of NMOS and PMOS devices. We have already discussed that V_{TH} can change due to process variations, which implies that the required β ratio for equal rise and fall times is also affected by process variations. Equal rise and fall times are desired, because under that condition, one gets highest frequency for a ring oscillator, short circuit power is decreased and the impact of PV on the circuit functionality is reduced [36]. Since β ratio for equal rise and fall times changes due to PV, a method that adaptively changes the β ratio by sensing



(a) β=1 (strong NMOS)

(b) β=6 (strong PMOS)

Fig. 6.17 Measured results on long inverter chains show the use of ABRM for salvaging failing chips

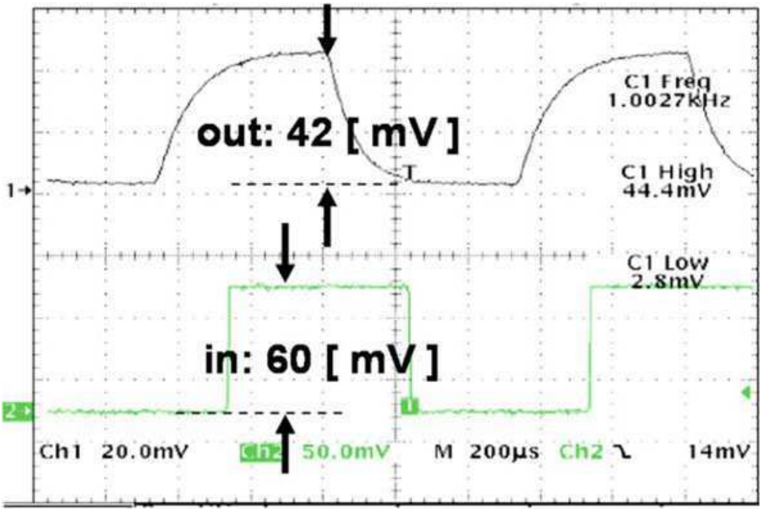


Fig. 6.18 Ultralow voltage operation on inverter chains showing a limiting case of operation of CMOS inverters

the original strengths of NMOS and PMOS is needed. This method called Adaptive Beta Ratio Modulation (ABRM) has been proposed in [36].

In [36], this technique uses body biasing to modulate β ratio. Three types of biases viz. forward body bias (FBB), reverse body bias (RBB), and zero body bias are applied. Logic threshold voltage (V_M) of an inverter is compared against two reference voltages V_{ref_1} and V_{ref_2}. If $V_M < V_{ref1}$, NMOS is stronger, and hence RBB is applied to increase V_{TH}. Alternatively, FBB can be applied to PMOS to decrease its V_{TH}, thus making it stronger. For the other case, when $V_M > V_{ref2}$, FBB is applied to NMOS. When $V_{ref1} < V_M < V_{ref2}$, a zero body bias is applied. Simulation and measurement results show that the technique leads to considerable reduction in the variation of logic threshold voltage.

Figure 6.17 shows the effect of ABRM on long inverter chains that emulate logic paths. While without ABRM, the output gets stuck at low or high under variation at low operating voltages; application of ABRM leads to correct functionality [36]. Measurement results shown in Fig. 6.18 demonstrate the operation of a 1000 chain inverter at voltages as low as 60 mV with an output swing of 42 mV.

6.7 Summary

This chapter provides an overview of the device, circuit, and architecture co-optimization required to provide minimum power dissipation for a target frequency of operation, in the subthreshold region. Several techniques at different levels of the design hierarchy have been discussed. This is an area of active and ongoing research, and interested readers are referred to references at the end of the chapter for further reading.

Acknowledgments The authors would like to acknowledge the contributions and helpful discussions with Prof Kaushik Roy, Mr. Sumeet Gupta, Mr. M. Hwang, and other members of the Nanoelectronics Research Lab in Purdue University.

References

1. Chandrakasan AP, Sheng S, Broderson RW (Apr 1992) Low-power CMOS digital design. IEEE J Solid-State Circuits 27:473–484
2. Soeleman H, Roy K, Paul BC (2000) Robust ultra-low power subthreshold DTMOS logic. In: IEEE international symposium on low power electronics (ISLPED), pp 94–96
3. Soeleman H, Roy K, Paul BC (2001) Robust Subthreshold logic for ultra-low power operation. IEEE Trans VLSI Syst 9(1):90–99
4. Wang A, Chandrakasan A (Jan 2005) A 180-mV subthreshold FET processor using a minimum energy design methodology. IEEE J Solid-State Circuit 40(1):310–319
5. Raychowdhury A, Paul BC, Bhunia S, Roy K (Nov 2005) Device/circuit/architecture co-design for ultra-low power digital sub-threshold operation. IEEE trans VLSI Syst 13: 1213–1224
6. Cannillo F, Toumazou C (2005) Nano-power subthreshold current-mode logic in sub-100 nm technologies. Electron Lett 41(23):1268–1269
7. Tajalli A, Brauer EJ, Leblebici Y, Vittoz E (Jul 2008) Subthreshold source-coupled logic circuits for ultra-low-power applications. IEEE J Solid-State Circuits (JSSC) 43(7):1699–1710
8. Nyathi J, Bero B (2006) Logic circuits operating in subthreshold voltages. In: The Proceeding of ISLPED international symposium on low power electronic design, pp 131–134
9. Elgharbawy W, Bayoumi M (2004) New bulk dynamic threshold NMOS schemes for low-energy subthreshold domino-like circuits. In: Proceeding of ISVLSI, pp 115–120
10. Thomas O, Amara A, Vladimirescu A (2003) Stability analysis of a 400 mV 4-transistor CMOS-SOI SRAM cell operated in subthreshold. In: Proceeding the Conference on Electron Device and Solid-State Circuits, pp 247–250
11. Chen J, Clark LT, Chen TH (Oct 2006) An ultra-low-power memory with a subthreshold power supply voltage..IEEE J Solid-State Circuits 41(10):2344–2353,
12. Hanson S, Seok M, Sylvester D, Blaauw D (Jan. 2008) Nanometer device scaling in subthreshold logic and SRAM. IEEE Trans Electron Devices 55(1):175–185
13. Calhoun BH, Wang A, Chandrakasan A (Sep. 2005) Modeling and sizing for minimum energy operation in subthreshold circuits. IEEE J Solid-State Circuits (JSSC) 40(9):1778–1786
14. Keane J, Eom H, Kim TH, Sapatnekar S, Kim C (May 2008) Stack sizing for optimal drivability in subthreshold circuits. IEEE Trans VLSI Syst 16(5):598–602
15. Melek LAP, Schneider MC, Galup-Montoro C (2004) Body-bias compensation technique for subthreshold CMOS static logic gates. In: The Proceeding of symposium On Integrated Circuits and System Design (SBCCI), pp 267–272
16. Chen J, Clark LT, Cao Y (2005) Robust design of high fan-in/out subthreshold circuits. In : Proceeding of international conference on Computer Design (ICCD), pp 405–410
17. Paul BC, Raychowdhury A, Roy K (Feb 2005) Device optimization for digital sub-threshold logic operation. IEEE Trans Electron Devices 52:237–247
18. Chakraborty S, Mallik A, Sarkar CK, Rao VR (Feb 2007) Impact of halo doping on the subthreshold performance of deep-submicrometer CMOS Devices and circuits for ultralow power analog/mixed-signal applications. IEEE Trans Electron Devices 54(2):241–248
19. Paul BC, Roy K (Feb 2008) Oxide thickness optimization for ultra-low power digital subthreshold operation. IEEE Trans Electron Devices 55(2):685–688,
20. Jayakumar N, Gamache B, Garg R, Khatri SP (2006) A PLA based asynchronous micropipelining approach for subthreshold circuit design. In: Proceeding of Design Automation Conference (DAC), pp 419–424

21. Chang IJ, Park SP, Roy K (Feb 2010) Exploring asynchronous design techniques for process-tolerant and energy-efficient subthreshold operation. IEEE J Solid-State Circuits (JSSC) 45(2): 401–410
22. Jorgenson RD et al (Feb 2010) Ultralow-power operation in subthreshold regimes applying clockless logic. Proc IEEE 98(2):299–314
23. Zhai B et al (Aug 2009) Energy-efficient subthreshold processor design. IEEE Trans VLSI Syst 17(8):1127–1137
24. Taur Y, Ning TH (1998) Fundamentals of modern VLSI devices. Cambridge University Press, New York, NY
25. Zhou X, Lim KY, Lim D (Jan 2000) A general approach to compact threshold voltage formulation based on 2D numerical simulation and experimental correlation for deep-submicron ULSI technology development [CMOS]. IEEE Trans Electron Devices 47(1):214–221
26. Kim CH-I, Soeleman H, Roy K (Dec 2003) Ultra-low-power DLMS adaptive filter for hearing aid applications. IEEE Trans VLSI Syst 11(6):1058–1067
27. Assaderaghi F et al (Mar 1997) Dynamic threshold-voltage MOSFET (DTMOS) for ultra-low voltage VLSI. IEEE Trans Electron Devices 44:414–422
28. Seok M, Hanson S, Lin Y-S, Foo Z, Kim D, Lee Y, Liu N, Sylvester D, Blaauw D (2008) The phoenix processor: a 30 pW platform for sensor applications. In: Symposium on VLSI circuits. Digest technical papers, 18–20 June, 2008, pp 188–189
29. Kim T-H, Liu J, Keane J, Kim CH (May 2008) Circuit techniques for ultra-low power subthreshold SRAMs. In: IEEE international symposium on Circuits and Systems, 2008, pp 2574–2577
30. Verma N, Chandrakasan AP (Jan 2008) A 256 kb 65 nm 8T Sub-threshold SRAM employing sense-amplifier redundancy. IEEE J Solid State Circuits 43(1):141–149
31. Ye Y, Borkar S, De V (Jun 1998) A new technique for standby leakage reduction in high performance circuits. In: Proceeding of IEEE symposium on VLSI circuits, pp 40–41
32. Gupta SK, Raychowdhury A, Roy K (2010) Digital computation in sub-threshold region for ultra-low power operation: a device-circuit-system co-design perspective. In: Proceedings of IEEE
33. Kulkarni JP, Kim K,Roy K (Oct 2007) A 160 mV robust schmitt trigger based subthreshold SRAM. IEEE J Solid-State Circuits 42(10):2303–2313
34. Kulkarni JP, Kim K, Park SP, Roy K (Jun 2008) Process variation tolerant SRAM for ultra-low voltage applications. In: Proceeding of design automation conference, pp 108–113
35. Chang IJ, Kim J-J, Park SP, Roy K (2008) A 32 kb 10T sub-threshold SRAM array with bit-interleaving and differential read scheme in 90 nm CMOS. In: IEEE International soli state circuits conference, pp 387–389
36. Roy K, Kulkarni JP, Hwang M-E (Jan 2008) Process tolerant ultralow voltage digital subthreshold design. In: IEEE Topical Meeting Silicon Monolithic Integrated Circuits in RF Systems, pp 42–45

Part III
System-Level Design Solutions

Chapter 7
Variation-Tolerant Microprocessor Architecture at Low Power

Meeta S. Gupta and Pradip Bose

Abstract This chapter focuses on the different techniques at the microarchitectural level to handle variations and to provide a variation-tolerant low-power design. An overview of the different sources of parameter variations is presented, and a discussion of the different models used at the architectural level to understand their effect on processor power and performance is provided. All sources of variations lead to timing overhead and uncertainty, but each kind of variation has different characteristics. Designers typically account for parameter variations by inserting conservative margins that guard against *worst-case* variation characteristics to guarantee functional correctness of the system under all operating conditions. However, such a conservative approach leads to significant performance degradation in the current technology and will worsen with technology scaling. This chapter presents alternative error-tolerant schemes to deal with these different sources of parameter variations. Error-tolerant schemes aim to run with nominal timing margins but without comprising robustness and correctness.

7.1 Introduction

In this chapter, we focus on chip (micro)architectural innovations related to variation-tolerant, low-power design. We consider several sources of variation, namely:

(a) Process variation in late CMOS era designs.
(b) Thermal hot spots and gradients.
(c) Voltage variations caused by inductive noise.
(d) Aging effects caused by device and interconnect wearout.

M.S. Gupta (✉)
IBM T.J. Watson Research Center, Yorktown Heights, NY, USA
e-mail: msgupta@us.ibm.com

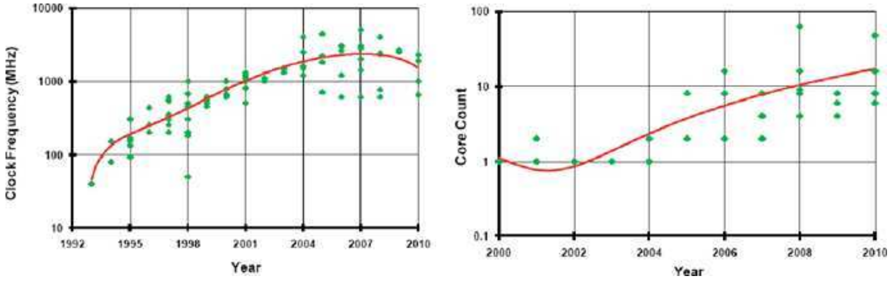
S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_7,

(e) Unanticipated variation (corruption) of microarchitectural state caused by soft errors, induced by energetic particle strikes from the chip package or the atmosphere.

In all or most of the above categories, there is an implied underlying causative agent: namely, the workload variability experienced by the chip hardware. In some cases (e.g., (a) above), the root cause is *not* the executing application workload; however, even in such instances, workload variability effects need to be modeled in order to quantify the end (or manifested) effect in terms of system reliability. In other cases (e.g., (c)) workload variability can be viewed to be the root cause that induces current variations, leading in turn to inductive noise issues. In some cases (e.g., (b) or (d)), there may be some baseline workload-independent hot spot, gradient and aging effects, with just "power and clocks on." However, the executing workload characteristics can (and usually do) provide additional "bias" that would cause more pronounced variability effects across the chip die. In particular, the time constants involved in the onset of observable aging-induced variabilities depend on the particular wearout mechanism *and* the workload characteristics. Regardless of the root underlying cause behind a manifested effect, the challenge for the modern processor chip microarchitect is to ensure that the early-stage design decisions incorporate appropriate features in the design that serve to provide robust functionality (and performance) at affordable power consumption.

7.1.1 Chip Microarchitecture Trends in the Power-Constrained Design Era

Before we delve into the details of variation-tolerant microarchitecture design, we provide a brief summary of the chip microarchitecture trends that industry has witnessed over the last decade, as a result of the "power wall" that the chip design community is very familiar with at this time. Figure 7.1 shows the trend of slowdown in the chip frequency growth (with attendant growth in per-chip core counts)



(a) (b)

Fig. 7.1 Processor frequency and core count trends [69]

as reported in the latest trend data (2010) published by the International Solid State Circuits Conference (ISSCC).

The frequency slowdown is caused by power density limits as far as affordable cooling solutions are concerned. Coincidentally, this has also been matched by a saturation of microarchitectural performance (quantified as instruction level parallelism (or ILP); measured typically as instructions per cycle (or IPC)) when it comes to a single core's performance growth over generations. As single cores hit ILP limits as well as frequency limits, the natural evolution, dictated by demand for ever-increasing chip performance, has been in the direction of increased number of processor cores within a single die [7]. The number and complexity of the processor cores depend on the particular product family, i.e., the balance of single-threaded and throughput performance demand that a particular market represents. There is also a certain trend of using ultra-efficient, special purpose accelerator "cores" interspersed with general purpose cores to create a heterogeneous mix that can perform well across both classes of workload demands, i.e., single-threaded ones, multi-threaded ones, and throughput-oriented ones. For high-end commercial servers, the emphasis is on throughput computing; nonetheless in this space, the demand for single-thread latency improvements also continues unabated. As such, for these systems, we usually see a trend toward heterogeneous or hybrid computing architectures at all levels of design: from chip-level microarchitectures upwards. For supercomputing systems [13] focused on high-performance computing (HPC), the strive to achieve power efficiency (in terms of gigaflops per watt) is so pronounced that individual chips usually end up using very simple processor cores, and parallelizing compilers extract enough parallelism from HPC codes to ensure delivery of high performance at affordable power. For unix servers targeted for a mix of commercial enterprise customers and HPC, we see the increasing use of high-ILP cores, supported by simultaneous multi-threading [28, 55] in an attempt to satisfy both market segments. Regardless of the target market space(s), the multi/many-core hybrid system design era has by no means eliminated the power wall! It has only enabled us to go beyond the paradigm of single core design, while maintaining an affordable power envelope for some time. But with the scaling of technology nodes in the late CMOS era, we are witnessing the effects of variability in numerous forms as already mentioned. This is causing renewed pressure on already strained power limits, since many of the sources of variability can be partially mitigated by increased operating chip supply voltage, but the latter causes power to increase!

A positive trend in chip design is the availability of embedded DRAM (EDRAM) technology. Processors like the IBM Blue Gene/L [13] or POWER7 [28] used EDRAM to build in many megabytes of on-chip cache to ease the pressure on performance. This translates to being able to rely less on frequency or per-core microarchitectural complexity growth, while adhering to targeted chip performance growth at manageable power. In addition, features like dynamic voltage and frequency scaling (DVFS) and power gating (PG) controlled by on- or off-chip power management controllers [28, 31] have enabled systems to grow in targeted performance within budgeted power envelopes. However, many of the

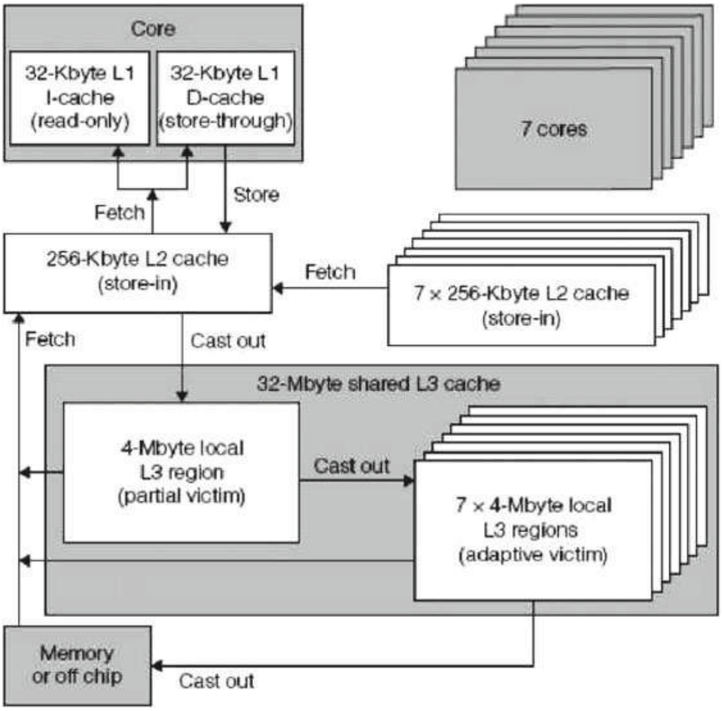

Core
32-Kbyte L1 I-cache (read-only)
32-Kbyte L1 D-cache (store-through)
7 cores
Fetch
Store
256-Kbyte L2 cache (store-in)
Fetch
7 × 256-Kbyte L2 cache (store-in)
Fetch
Cast out
32-Mbyte shared L3 cache
4-Mbyte local L3 region (partial victim)
Cast out
7 × 4-Mbyte local L3 regions (adaptive victim)
Memory or off chip
Cast out

Fig. 7.2 Power7 multi-core chip: cache hierarchy (as reported by Kalla et al. [28]). Each core has a 32 Kbyte instruction cache and a 32 Kbyte data cache. A 256-byte L2 cache and a 4-Mbyte local region of a 32-Mbyte shared L3 cache

aggressive power managements features (e.g., DVFS and PG) are causative agents behind problems like inductive noise on the voltage rails, triggered by workload variations. Also, lower voltage operation under DVFS enhances risk of particle-induced soft error upset events that cause potentially damaging transient alterations to the microarchitecural state of the machine.

Figure 7.2 shows the high-level block diagram of a current-generation multi-core processor architecture (POWER7^{TM} [28]), with an emphasis on depicting the on-chip cache hierarchy. POWER7 is an eight-core processor chip, with four-way simultaneous multi-threading (SMT) provided for each core. The processor core floorplan (as reported in [28]) has six main regions: the instruction fetch unit (IFU), the instruction sequencing unit (ISU), the load-store unit (LSU), the fixed point unit (FXU), the vector and scalar unit (VSU) which includes the scalar double precision floating point engine as well as full support for the VMX (SIMD) extension of the PowerPC architecture, and the decimal floating point unit (DFU). The IFU includes both a condition register logic unit (CRU) and a branch execution unit (CRU). In a given cycle, each POWER7 core can fetch up to eight instructions, decode and dispatch up to six instructions, and issue/execute up to eight instructions. There are a total of 12 execution units within each core: two integer (fixed point) units, four double precision floating point unit (FPU) pipelines, one vector (SIMD) unit, one branch execution unit (BRU), one condition register logic unit (CRU), and one decimal floating point unit pipeline. The two load-store pipes can also execute simple fixed point operations.

Process variability affects combinational logic paths (between pipeline stage latch boundaries) and latch speeds, and in current designs, the mitigation solution is worst-case design by using a big enough timing guard band. For on-chip caches made out of SRAM arrays, there is a more basic problem of cell stability as supply voltages shrink downwards. SRAMs use minimum-sized devices and are affected the most by intra-die variations in device dimensions and threshold voltages. Such variations result in mismatch of device strengths (within the cross-coupled inverter element of an SRAM cell), and this can render read operations unstable: i.e., they can flip the value stored in the cell. As supply voltages dip, the variability gets worse and this affects reliable operation of SRAMs. This poses a big challenge for scaling classical 6T SRAM cell-based caches to operate at lower voltages in scaled technology roadmaps. Many current generation processor chips are forced to use special, elevated supply voltage rails for SRAM cache macros.

7.1.1.1 The Impact of Variability on Multi-core Architectures

Power and ILP constraints have launched us into the multi-core era. What are the additional trends in architecture that variability-related constraints may be forcing us into? Let us briefly consider the possibilities:

- Intra-die process variation affects design in multiple ways. For example:

  (a) Different cores on the chip may have different voltage–frequency (v, f) characteristics. Thus, for a common supply voltage, each core could manifest a different (f_{min},f_{max}) operating range as far as the clock frequency is concerned. Similarly the (V_{min}, V_{max}) operating range from a basic circuit reliability viewpoint could be different for each core. One way to cope with such variability is to design for the worst case. In other words, the chip-level operating ranges could be constrained to cater to the slowest core or the weakest core to preserve reliable functionality. However, with the steady increase in variability, such a design practice would severely constrain the effective chip performance growth over technology generations. Hence, alternative paradigms in architecture that would allow chips to operate under multiple voltage and frequency domains [24, 28, 31, 52] are likely to become mainstream. Asynchronous bus interfaces may be required to support such paradigms, thereby adding to pre-silicon verification complexity.

  (b) Different cores (or regions) on the chip may age differently. This would cause the core(s) aging most rapidly to determine the mean time to failure (MTTF), and this may be much smaller than the nominal design specification. New generation architectures (e.g., [33]) may therefore have to invest into on-chip aging sensors that allow (small time-constant) dynamic adaptation of per-core (v, f) operating points or other microarchitectural knobs to ensure uniform aging rates across the die. Wear-leveling [45, 54] and variation-aware task scheduling techniques [32] are alternate mechanisms that bear promise in this context. Spare cores (and associated resources) may become commonplace, even in high-volume (low-end) server offerings.

- Thermal hot spots and gradients affect chip lifetime reliability:

  (a) A pronounced, persistent hot spot in certain core, or in a particular unit within a core, may cause an early wearout-related failure. As such, concepts like activity migration or thermal-aware task scheduling [10, 32] may find their way into real hardware–software solutions for future systems.

  (b) Across-die thermal gradients and dynamic thermal cycling cause premature chip failures as well. Again, dynamic load redistribution coupled with online thermal monitoring might be needed to mitigate such effects in future systems.

- Voltage variations may cause circuit malfunctions. For example:

  (a) Voltage dips [15, 29] due to inductive noise on the supply rails may cause some timing paths to fail at the nominal frequency of the processor, leading to catastrophic failure or even silent data corruption. Therefore, built-in feedback control loops to adjust down the clock frequency in response to an impending voltage droop may become routine in future chip designs. Necessary sensors, working in conjunction with digital PLL circuitry, would be needed in such chip architectures. Pre-silicon verification complexities will be a major challenge in these scenarios.

  (b) Voltage surges, caused by inductive noise and workload variations, are also a reliability concern [4, 15, 36, 60, 63], since they can accelerate specific failure mechanisms such as device oxide breakdown. In addition to decoupling capacitors, reliance on feedback control loop-based architectures may again become necessary in order to ensure robust performance growth over chip design generations.

- Aging and wearout effects, even if uniformly spread across the chip, affect reliability:

  (a) Even if all cores wear out evenly, the first onset of effects like NBTI or electromigration [21, 22, 39, 64] may manifest as intermittent failures at random across the chip die. Weak spots within individual cores and interconnects caused by random variations in the manufacturing process are the most vulnerable. The most defective chips get eliminated by chip burn-in methods, prior to shipment. Yet, the process of burn-in itself can worsen predisposition to failure; so, even if the chips pass the burn-in process, there will be latent weak spots that can cause premature failure in the field. The fact that burn-in methods are becoming progressively difficult, due to leakage power dominance and thermal runaway effects, is causing a further exacerbation of this problem. So again, architectures that provision for spares at the core, sub-core and cache (SRAM) level [54] and allow for redundant communication paths [40] will pave the way for robust operation in the face of such random wearout manifestations across the chip die.

(b) Mixed technology architectural design offers a possible solution approach to combat the incidence of random defects. For example, the DIVA approach [2] advocates the use of simple checker cores (built using older, more robust technology) to augment regular, high performance cores in order to achieve the tolerance against unpredictable occurrence of faults (transient, intermittent, or permanent) in future designs.

- Soft errors, caused by high energy particle strikes present a major source of uncertainty and variability, with increasing concern in late CMOS era design [4, 37]. Examples of state corruption and attendant mitigation efforts are:

  (a) The biggest concern in this particular domain of uncertainty is that of silent data corruption, i.e., the potential corruption of program output data that fails to be detected. Server-class, high-end microprocessors often invest into the use of redundant cores or coding techniques to ensure detection (and in cases, recovery-based corrective measures). In view of power and area concerns, the design community today is looking for lowest overhead software–hardware architectural solutions [37] that are capable of tolerating errors, without blowing the tight power budget.

  (b) Detectable, but unrecoverable errors that lead to machine checkstop are also of concern. Here, through the use of virtualized system software, system availability can be maintained, without loss of data in most cases. In high-end systems [25], the uncorrupted machine checkpoint state of a failed processor can be migrated to another core to preserve seamless computation without visible downtime from the end-user viewpoint.

7.1.2 Summary

In this introductory section, the general technological trends that motivate the development of variation-tolerant, low-power processor architectures have been presented. Some key examples of the sources of variation and the effect they have at the system architecture level were discussed. At the same time, viable solution approaches of current promise were referred to briefly. Later, in Section 7.4, we cover the architectural solution approaches in more detail and variety.

7.2 Modeling Parameter Variations

Parameter variations can be broadly classified into device variations incurred due to imperfections in the manufacturing process and environmental variations due to fluctuations in on-die temperature and supply voltage. Collectively, these parameter variations greatly affect the speed of circuits in a chip; delay paths may slow down or speed up due to these variations. The traditional approach to deal with parameter variations has been to over-design the processor based on the most pessimistic operating conditions to allow for worst-case variations. As the gap between nominal and worst-case operating conditions in modern microprocessor designs grow, the inefficiencies of worst-case design are too large to ignore. Recognizing the inefficiencies in such a design style, researchers have begun to propose architecture-level solutions that address worst-case conditions.

An important first step to design solutions is to understand how parameter variation affects delay of critical paths in high-performance processors. The delay of a gate (Equation (7.1)) is a function of process variation plus run-time variations (voltage and temperature). Process variation affects threshold voltage and effective gate length of a gate, voltage variation affects supply voltage, and temperature variation affects leakage power and threshold voltage. This section presents the models used to understand the effects of different parameter variations. An accurate model of parameter variations needs to represent all the three main sources of variations (PVT) to evaluate their effect on processor power and performance.

7.2.1 Process Variations

Process variation is mainly caused by fluctuations in device channel dimensions and dopant concentrations. Gate length variation can change the effective driving capability of the transistor, as well as the threshold voltage, due to the short channel effect. Random dopant variations can also change the threshold voltage of the device. Process variation mainly affects the threshold voltage (V_{th}) and effective gate length (L_{eff}) of the transistors. These two parameters in turn affect the delay of the gate (T_{g}) and the leakage power of the gate (P_{leakage}). These two parameters along with the dynamic power of a gate (P_{dynamic}) are given by Equations (7.1), (7.2), (7.3), where V, T, C and f are the supply voltage, temperature, capacitance, and frequency, respectively, while μ and α are process parameters and q and k are physical constants.

$$T_{\text{g}} \propto \frac{L_{\text{eff}} V}{\mu (V - V_{\text{th}})^{\alpha}} \tag{7.1}$$

$$P_{\text{dynamic}} \propto C V^2 f \tag{7.2}$$

$$P_{\text{leakage}} \propto V T^2 e^{-q V_{\text{th}} / kT} \tag{7.3}$$

Architectural studies mainly rely on statistical models of variation driven by values projected by the ITRS [23, 34]. Liang et al. [34] present a Monte-Carlo-based variation model which use first-order approximation model to account for the additional delay caused by parameter fluctuations. VARIUS model [51] is widely used to characterize the effect of process variations on the timing of different structures in a microprocessor. To model systematic variation, the chip is divided into a grid. Each grid point is, given one value of the systematic component of the parameter, assumed to have a normal distribution with $\mu = 0$ and standard deviation σ_{sys}. Systematic variation is also characterized by a spatial correlation, so that adjacent areas on a chip have roughly the same systematic component values. The spatial correlation between two points x and y is expressed as $\rho(r)$, where $r =| \overrightarrow{x} - \overrightarrow{y} |$. To determine how $\rho(r)$ changes from $\rho(0) = 1$ to $\rho(\infty) = 0$ as r increases, the spherical function is used. The distance at which the function converges to zero is when there is no significant correlation between two transistors. Such distance is called ϕ. Random variation occurs at the level of individual transistors. It is modeled analytically with a normal distribution with $\mu = 0$ and standard deviation σ_{ran}. Since the random and systematic components are normally distributed and independent, their effects are additive, and the total σ is $\sqrt{\sigma_{\text{ran}}^2 + \sigma_{\text{sys}}^2}$. In the model, V_{th} and L_{eff} have different values of σ.

7.2.2 Temperature Variations

Temperature is a function of the total power of the system, including both the dynamic and leakage power. Equation (7.4) highlights T as a function of the temperature of the common heat sink (T_{H}), the thermal resistance of the subsystem (R_{th}) and the total power of the system (P_{dynamic} and P_{leakage}). Variation in temperature has a slower time constant (in the order of a few tens of thousands of cycles) as compared to changes in voltage (in the order of a few cycles). HotSpot [56] is an accurate yet fast thermal model for architectural studies. HotSpot exploits the well-known duality between thermal and electrical systems [30]. Figure 7.3 depicts the HotSpot model for an architecture with three units. Each unit on the chip is modeled as a heat (power) dissipator; each underlying region of silicon is modeled as part of the RC circuit, with several RC elements representing lateral and thermal heat flow; and the package and convection contribute additional RC elements. HotSpot generates the equivalent RC circuit automatically based on some basic information about the package and desired floorplan, and, supplied with power dissipations over any chosen time step, computes temperatures at the center of each block of interest. HotSpot provides the steady-state and transient characteristics of a given workload. Operating temperature from HotSpot feeds back into the power and delay models (Equations (7.2), (7.3), (7.4), (7.5), (7.6)) to accurately capture the temperature transients to account for run-time effects of temperature.

$$T = T_{\text{H}} + R_{\text{th}} x (P_{\text{dynamic}} + P_{\text{leakage}}) \tag{7.4}$$

$$\mu \propto T^{-1.5} \tag{7.5}$$

$$V_{\text{th}} = V_{\text{th0}} - k_1 (T - T_0) \tag{7.6}$$

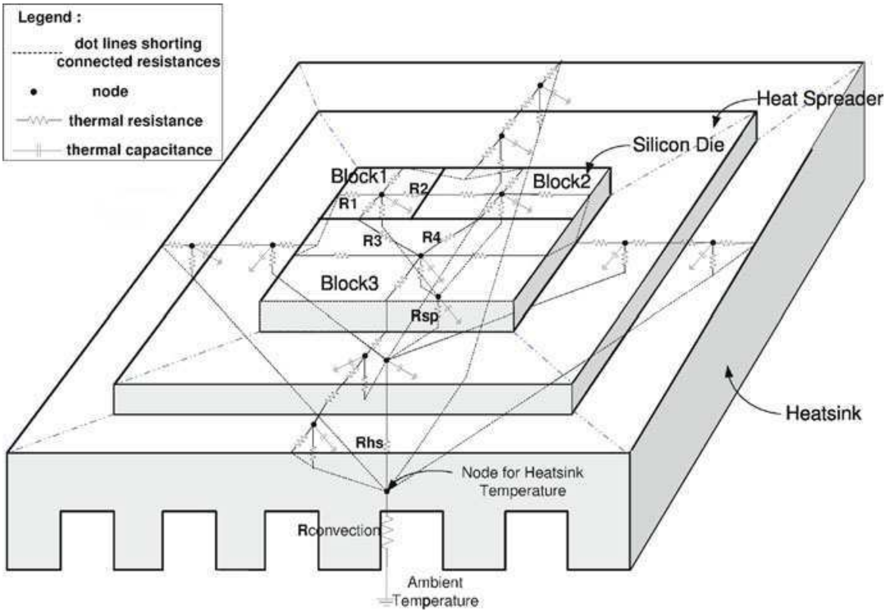


Fig. 7.3 Example HotSpot RC model. The assumed floorplan has three architectural units, a heat spreader, and a heat sink

7.2.3 Voltage Variations

Sudden current swings due to activity fluctuations in processors, when coupled with parasitic resistances and inductances in the power delivery subsystem, give rise to large voltage swings. Equation (7.7) highlights how current fluctuations interact with the impedance of the system (represented by *Z*). A decrease in supply voltage leads to an increase in the delay of the gates (Equation (7.1)). Voltage also impacts both the dynamic power and leakage power of the system (Equations (7.2) and (7.3), respectively).

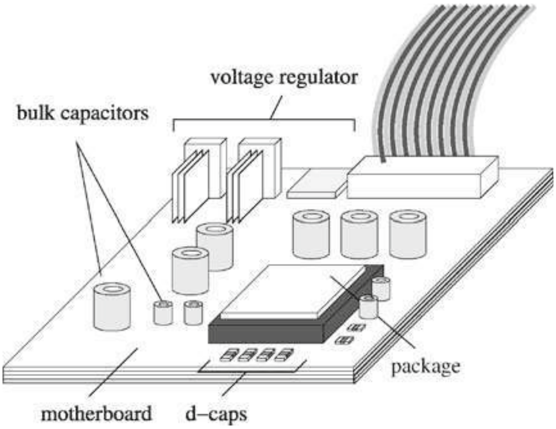
$$V = V_{\text{dd}} - V_{\text{drop}} \tag{7.7}$$

$$V_{\text{drop}} = Z \times I_{\text{instantaneous}} \tag{7.8}$$

$$I_{\text{instantaneous}} = P_{\text{leakage}} + P_{\text{dynamic}} V_{\text{dd}} \tag{7.9}$$

Figure 7.4 provides an overview of a generic power-delivery subsystem including the VRM, motherboard, package, and off-chip decoupling capacitors. The off-chip network includes the motherboard, package, and off-chip decoupling capacitors and parasitic inductances, modeled via a ladder RLC network. The bulk capacitance on

Fig. 7.4 Overview of microprocessor power delivery subsystem. This figure depicts the main components associated with the power-delivery subsystem of a processor



the PC board and the package can be modeled as an effective capacitance and effective series resistance. Voltage regulator modules (VRM) typically have response frequencies in the sub-MHz range, which is much lower than the challenging higher frequencies associated with the entire power-delivery network. For simplicity, the power supply is modeled as a fixed voltage source, which is scaled with respect to the average current draw to deliver VDD at the bump nodes, mimicking the feedback loop associated with the VRM. The electrical properties of these network elements determine what impact current variations at different frequencies have on chip-level voltage. The impedance profile of a power-delivery subsystem describes the frequency sensitivity of the distribution network. An example impedance plot of a PentiumIV processor is shown in Fig. 7.5. The mid-frequency resonance peak is

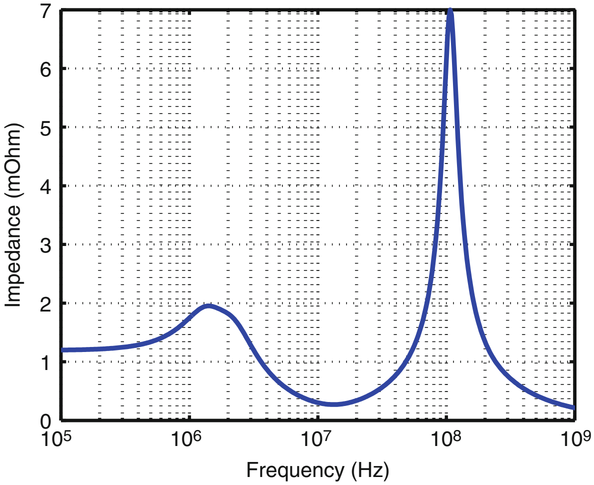


Fig. 7.5 Example impedance plot: This plot depicts an impedance plot modeled for a PentiumIV package, with mid-frequency resonance at 100 MHz and a peak impedance of 7 mΩ

the dominant property responsible for inductive noise problem. The mid-frequency resonance peak shown at 100 MHz in Fig. 7.5 is mainly due to the interaction of the package inductance and decoupling capacitance.

Since voltage variations are strongly coupled to the characteristics of the underlying power delivery subsystem, it is important to have good models for processor activity, power consumption, and the power delivery subsystem. This section explores the three different methods of modeling voltage noise: a detailed grid model, a impulse-response-based model, and a sparse grid model.

7.2.3.1 Distributed Grid Model

A good model of the power-delivery subsystem needs to capture the characteristic mid-frequency resonance, transients related to board and package interfaces, and localized on-chip voltage variations. A distributed model of the power-delivery subsystem provides good accuracy but at a simulation cost, making it appropriate for feedback driven experiments where a circuit solver interfaces with architectural level simulation for few thousands of cycles as opposed to longer simulations of millions of cycles.

Figure 7.6 presents an example of a detailed model of the power-delivery system (PDS) with a distributed on-chip power-supply grid. Figure 7.7 illustrates the distributed on-chip grid model used. The model assumes a flip chip package design, which is the frequent design choice for high-performance processors. The package connects to the chip through discrete controlled collapse chip connection (C4) bumps. The C4 bumps are modeled as parallel connections (via RL pairs) that connect the grid to the off-chip network, with each grid point having a bump connection. This model could be easily adapted to capture the behavior of a bond wire package instead. The on-chip grid itself is modeled as an RL network. The evenly distributed

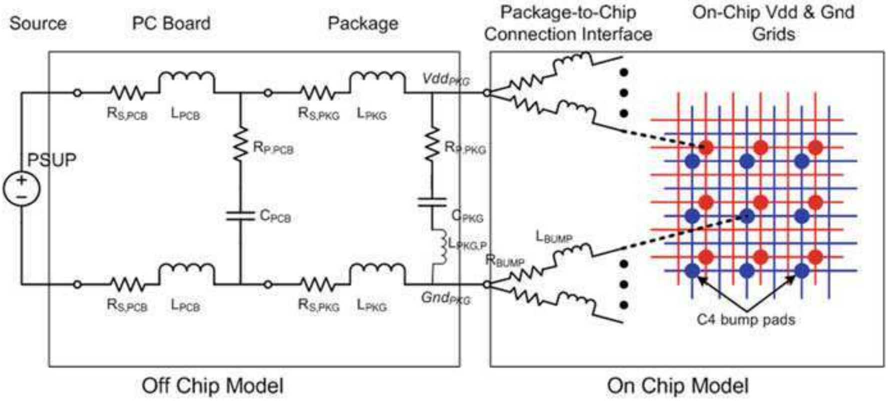


Fig. 7.6 Model of a power-delivery system: this figure shows the parasitics of the package, the package-to-chip interface, and the on-chip grid

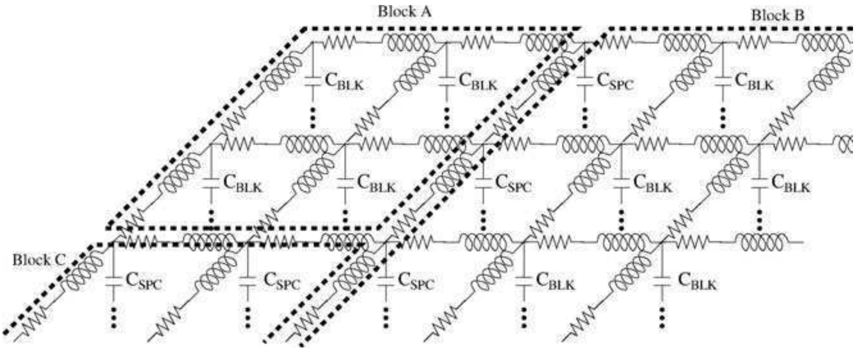


Fig. 7.7 On-die distributed grid model: this figure illustrates the parasitics of the on-chip grid model including the additional decoupling capacitance on the core

on-chip capacitance between the VDD and GND grids is modeled in two ways—C_{spc} represents the decoupling capacitance placed in the free space between functional units and C_{blk} represents the intrinsic parasitic capacitance of the functional units. In contrast, an on-chip lumped model would consist of a single RLC network connected across the package-to-chip interface. When scaling the size of the grid, the values in the RLC network are scaled proportionally. The amount R and L is scaled by the change in length of each grid segment, and the total capacitance between the power and ground planes is re-divided among the grid nodes. This yields a closely matching impulse response as the number of grid nodes is increased.

Accuracy of the power-delivery model is strongly dependent on the grid resolution of the on-chip distributed grid. A finer distributed grid should be able to give finer sampling of the voltage variations across the die, and hence would give a more accurate depiction of the variation across the chip. A coarser resolution of the grid fails to capture the finer variations across the chip. However, simulation speed is a critical issue if localized power delivery models must interface with architectural power and performance simulators. A fast circuit solver, based on preconditioned Krylov subspace iterative methods [9], that utilizes a SPICE netlist of the entire power-delivery network and per block current profiles to simulate on-die voltages can be embedded in cycle accurate simulators for architecture and system level studies. Table 7.1 outlines the speed of the lumped model and various grid models for a 100 K cycle sample run with a detailed power/performance model for a high-performance microprocessor.

Table 7.1 Simulation Speed of using a Distributed Grid Model: This table presents the simulation times of various grid sizes (100 K cycles)

Grid Size	1 × 1	2 × 2	4 × 4	6 × 6	8 × 8	12 × 12	16 × 16	20 × 20	24 × 24
Speed (s)	5	28	174	493	1070	1423	1830	4620	7280

7.2.3.2 Impulse-Response-Based Model

While the PDS of a given microprocessor is a complex system consisting of several different components (e.g., voltage regulator module, package, on-die capacitors, etc.) [3, 11], a simplified second-order lumped model [20, 57] can adequately capture its resonance characteristics with impedance peaking in the mid-frequency range of 50–200 MHz and can be reasonably modeled as an under-damped second-order linear system [20] as described by Equation 7.10.

$$a\frac{\mathrm{d}^2}{\mathrm{d}t^2}y(t) + b\frac{\mathrm{d}}{\mathrm{d}t}y(t) + cy(t) = f(t) \tag{7.10}$$

Ideally, the supply voltage across a processor should be constant. However, due to dynamic current fluctuations and the non-zero impedance of the PDS, large voltage fluctuations can occur. One way to characterize voltage variations is by convolving the instantaneous current profile of the microprocessor with the impulse response of the PDS (Equation (7.11)).

$$v(t) = i(t) * h(t) \tag{7.11}$$

Most of the earlier works that seeks to address voltage noise at the architectural level uses a simplified second-order lumped model [20, 57], which captures the mid-frequency response of the system. However, such a model fails to capture within-die spatial variation of voltage. While detailed grid models have also been proposed and used, the large number of nodes lead to prohibitively high simulation times; hence we explored an intermediate model described below.

7.2.3.3 Sparse Grid Model

A detailed distributed grid model suffers from slow simulation time, of the order of few hours per million instruction, making it impossible to use such a model for detailed workload characterization. To deal with the simulation time complexity of the model, Gupta et al. propose a simpler and faster on-chip model to retain the per-unit or per-block characteristics [19]. The simulation time of such a model is approximately 1 hr/100 M instructions, which is of the order of magnitude faster than the detailed grid model. By appropriately scaling R, L, and C of each unit's power grid with respect to area, this model enables relatively fast simulations while maintaining accuracy that closely matches a detailed grid model. The R, L, and C for any unit will scale as given in Equation (7.12), where A_{unit} represents the area of the unit and A_{core} represents the area of the core. R_{l}, L_{l}, and C_{l} correspond to values found in a lumped model.

$$R = R_{\mathrm{l}} * \frac{A_{\mathrm{core}}}{A_{\mathrm{unit}}} \tag{7.12}$$

$$L = L_1 * \frac{A_{\text{core}}}{A_{\text{unit}}} \tag{7.13}$$

$$C = C_1 * \frac{A_{\text{unit}}}{A_{\text{core}}} \tag{7.14}$$

7.3 Modeling Aging-Induced Variations

In this section, we focus on the topic of aging-induced variations and the attendant challenge of modeling the effects at the (micro)architectural level. In general, aging occurs due to constant activity or use of the compute-communicate-store circuits in the processor. To the extent that the workload is ever-changing and non-uniform and that there are manufacturing-time dimensional variations across the die, the various chip regions experience non-uniform wearout. Thus, given enough time, ultimately one region would wear out the fastest and cause the full chip to fail. In order to model the wearout phenomenon for each class of aging, three aspects of variation need to be modeled: (a) the native variation of device and interconnect dimensions after the chip is manufactured; (b) the particular wearout mechanism modeled as a function of utilization (activity) and bias, operating environment (voltage, temperature, etc.), technology parameters, and time; and (c) variation in the input workload that determines the activity profile across chip resources. We will mainly be dwelling on aspects (b) and (c) in this particular section.

7.3.1 BTI and HCI

The aging mechanisms in nanoscale CMOS gradually reduce the I_{ON} current of active devices over time. This results in slowing down the corresponding circuits in terms of propagation delay of signals. As aging progresses, eventually, such slowdown can cause timing failures that lead to circuit failure. It is widely accepted that Bias Temperature Instability (BTI) and Hot Carrier Injection (HCI) are the two major physical phenomena that lead to transistor aging [21]. BTI in a PMOS device is termed NBTI, because the gate of the transistor is negatively biased (relative to the source). Similarly, BTI for an NMOS device is termed PBTI.

BTI phenomena cause the threshold voltage of the device to increase, thereby causing a reduction of its I_{ON} current. Since such aging is more severe in the case of NBTI [21], we limit our attention to NBTI when explaining the basic mechanism and how one can model it at the circuit and system level. NBTI is caused by interface states that exist at the junction of the silicon channel and the oxide layer (which is typically today a high-k dielectric material with a metal gate terminal), as depicted in Fig. 7.8a. Broken or dangling Si bonds, present at the imperfect interface separating the silicon and the dielectric, are initially repaired using hydrogen atoms

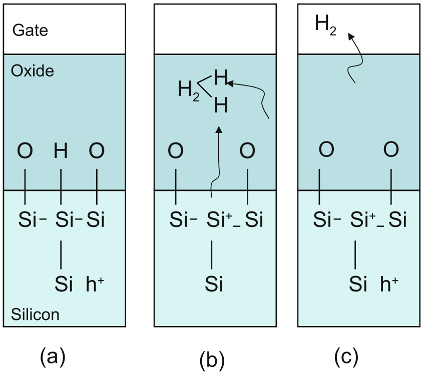


Fig. 7.8 NBTI chemical mechanism in PMOS: (**a**) hole induction in channel breaks Si–H bonds at Si–Oxide interface; (**b**) released hydrogen atoms diffuse into the oxide and form H_2 molecules; (**c**) a small fraction of these molecules diffuse and get trapped into the metal gate layer causing a positive charge residue at the Si–Oxide interface

that are prevalent during the chip fabrication process (e.g., via the hydrogen annealing steps). Under negative gate bias conditions, holes are induced into the channel, and these can break apart many of the weak Si–H bonds. As a result, hydrogen atoms released by broken Si–H bonds migrate of diffuse into the dielectric region (Fig. 7.8b) or may even get embedded permanently into the metal gate (Fig. 7.8c). Once the negative bias is removed, the majority of the hydrogen atoms migrate back to their original positions in bonding back with the Si atoms. However some, especially those embedded in the metal gate, do not make it back. Consequently, a residual small positive charge appears at the silicon–dielectric interface; and this can accumulate over time to start causing an upward shift of the threshold voltage of the PMOS device.

Hot Carrier Injection (HCI) is another key mechanism that contributes to device aging in the late CMOS era. The "hot carriers" are either the holes or the electrons that gain high kinetic energy after being accelerated by a strong lateral electric field. Due to their high energy, these hot carriers can get injected and trapped at the Si–oxide interface layer forming a small space charge, which expands gradually over time as more charges are trapped. This slowly expanding space charge causes gradual device slowdown due to increased threshold voltage.

7.3.2 Electromigration

This failure mechanism is well understood and extensive research has been performed by the material science and semiconductor community on modeling and understanding its effects [36]. Electromigration in processor interconnects is due to the mass transport of conductor metal atoms in the interconnects. Sites of metal atom depletion can lead to increased resistance and open circuits. At the site of metal atom pileup, extrusions can form, causing shorts between adjacent lines. The model for mean time to failure (MTTF) due to electromigration [60] is

$$\mathrm{MTTF}_{\mathrm{EM}} = K.(J)^{-n}\mathrm{e}^{\frac{E_\mathrm{a}}{kT}} \tag{7.15}$$

where J is the current density in the interconnect, E_a is the activation energy, k is Boltzmann's constant and T is the absolute temperature in degrees Kelvin. E_a and n are constants that depend on the type of interconnect metal used; K is just a constant of proportionality expressed in appropriate units.

At the microarchitectural level of design, structures are modeled at the granularity of entire functional units (e.g., instruction decode and dispatch unit, IDU). Utilizations for each modeled microarchitectural unit can be obtained from a microarchitectural, cycle-accurate performance simulator, driven by target workload traces. Effective current density (J) over a modeled unit can be computed as an average abstraction, from the known utilization and the maximum rated current density sustainable by interconnects in that unit. Such a modeling approach is used by the microarchitecture-level RAMP simulator developed by researchers at the University of Illinois and IBM Research [60].

7.3.3 Time-Dependent Dielectric Breakdown (TDDB)

Loosely known as oxide breakdown, TDDB is another well-understood failure mechanism in CMOS devices. The gate oxide (or dielectric) wears down with time, and fails when a conductive path forms in the dielectric [21, 60, 64]. The model for the oxide breakdown related MTTF used in RAMP is based on experimental work performed by Wu et al. at IBM [68]:

$$\mathrm{MTTF}_{\mathrm{TDDB}} \propto \left(\frac{1}{V}\right)^{a-bT} \mathrm{e}^{\frac{X+\frac{Y}{T}+ZT}{kT}} \tag{7.16}$$

Where T is the absolute temperature in degrees Kelvin; a, b, X, Y and Z are fitting parameters; and V is the voltage. Based on the experimentally measured data collected by Wu et al., RAMP [60] uses the following TDDB-related model parameters: $a = 78, b = -0.081, X = 0.759\mathrm{ev}, Y = -66.8\,\mathrm{evK}$ and $Z = 8.37\mathrm{e} - 4\mathrm{ev/K}$.

7.3.4 Improved Modeling Fidelity and Flexibility via STAR

In subsequent modeling work for multi-core processor architectures, Shin et al. [53] used an interesting abstraction at the higher structural level, while achieving high accuracy in terms of low-level modeling fidelity. In RAMP, a uniform (average) activity or utilization per modeled unit was assumed. In the STAR model used by Shin et al., a reference circuit was used to quantify the failure rate (measured in FITs) precisely in terms of the technology and circuit low-level parameters. The reference circuit was constructed systematically, in order to capture a representative, and accentuated instance of when the particular failure mechanism of interest would manifest. Then, the higher level structure's FIT rate could be formulated in terms of FORCs (FIT of a reference circuit), abstracting away the detailed circuit-level failure mechanisms completely. In [53], the authors show how this method of modeling can be applied to various structures (e.g., SRAMs and register files) within a modern multi-core processor in order to project the mean time to failure, under various aging mechanisms like electromigration, TDDB, and NBTI.

7.4 Designing for Variation-Tolerance

System architecture-level solution approaches to variation tolerance may be broadly classified into the following categories: worst-case design, where timing margins and spatial (dimensional) design rules are made extremely conservative in order to ensure robust operation under a wide range of variability. This approach usually leads to significant performance shortfall in the current technology regime, and would even tend to make the basic benefit of moving to future technology nodes quite questionable. Design for error tolerance, where nominal design rules and timing margins are adopted without sacrificing performance targets of the new generation product; but, the system architecture is designed from the outset to tolerate errors that may arise from known sources of variation.

In this chapter, the interest is obviously in the second category. Within the second category of error-tolerant system architecture paradigms, there are again two broad principles that are used: (a) spatial redundancy and (b) temporal redundancy. Quite often, a particular system product may include elements of both (a) and (b) to provide cost-effective error tolerance, without compromising target performance goals. Classical system reliability approaches at the mainframe end have been known to use spatial redundancy in the form of core duplication (e.g., [58]). In such solutions, dual cores operate in lock-step (with shared level-1 or L1 caching) and compare results on a cycle-by-cycle basis. If the results match, the validated update to the register state is copied into an ECC-protected checkpoint area of the chip. The checkpoint area always contains a "golden" state of the machine, from which execution can be restarted if the dual cores detect a mismatch in computed results on any given cycle. The L1 cache is usually parity protected, with write-through to the L2 cache, which is ECC protected. In such a dual-core design, the area and power overhead relative to a single-core chip baseline is roughly 35–40; design approaches are generally not affordable, even at the high-end mainframe or server space. Also, the above dual-core redundancy approach covers the case where one core is in error due to a transient or permanent fault, but the other core is unaffected. This may work well for intra-die variations (due to process, voltage, or temperature) that cause a transient error affecting only one of the dual cores; but if the same error affects both cores identically (as in the case of some inter-die variations, for example) then this solution approach would not work in all error scenarios anyway.

One of the current solution approaches to guard against soft (or transient) errors is the pervasive use of so-called hardened latches (e.g., [5]). This can incur about a 25% area overhead, but has the advantage of zero additional complexity at the microarchitectural level. However, the coverage or protection against all sources of variation-induced errors is not guaranteed by hardening of latches alone. If the computing logic is error-prone, hardening the receiving latches will not be able to ensure correctness at the system level. Another solution approach, typified by the IBM POWER6 processor [48], is to move away from the dual-core redundancy, but to augment each core with enough error detectors (e.g., parity checkers) to ensure an acceptably high error detection coverage (for a given fault model). On detecting an error, computation is restarted from an ECC-protected checkpoint state, which is updated every cycle as in the prior mainframe-class processors. This approach can cut down the area overhead from nearly 40 [58]) down to as little as 12% or so. However, each core design is now more complicated, because of parity generators and checkers, routing of error detection flags to the recovery unit, and for selecting the right number and location of error checkers to ensure maximal coverage at lowest area and timing complexity. Of course, even in POWER6-style designs, a careful balance across use of hardened latches and error detectors may be needed to maximize error coverage, while minimizing area overhead. Incidentally, in POWER7 [28], an explicit checkpoint array and recovery unit have been avoided, while preserving the hardware checkpoint-restart capability available in POWER6. This is possible, since POWER7 is an out-of-order processor, so the rename buffers provide a means for holding temporary (uncommitted state), which can be discarded during a restart phase similar in a manner to recovering from a branch misprediction. The architected register state is ECC protected and serves at the golden checkpoint at any given instant of machine operation.

In the following sub-sections, specific other point solutions to the problem of tolerating variation-induced errors are discussed. These are recent research advances that are not necessarily reflected in actual products yet.

7.4.1 Process-Variation Aware Microarchitecture Design

Because of the inherent randomness associated with process variation, it can be difficult to predict the speed of individual circuit blocks before chip fabrication. While statistical timing techniques can help designers understand the extent of variations and suggest ways to best tune paths for logic structures [1], post-fabrication delay-path variations cannot be entirely avoided. Post-fabrication tuning solutions can provide an effective approach to address process variations. Existing solutions can be broadly divided into techniques based on adaptive body bias (ABB) and adaptive pipeline latency. ABB relies on tuning the body voltage of a transistor to manipulate its threshold voltage. This can be used to increase the speed of devices at the cost of increased leakage power [66]. At the architectural level, researchers have shown that ABB is most effective when applied at the granularity of microarchitectural blocks [65]. While ABB techniques are attractive, they are only applicable to bulk CMOS processes (e.g., ABB will not work with SOI and triple-gate processes being adopted by industry).

Ndai et al. [38] present a *variation-aware* scheduling methodology to deal with within-die variations. The technique sets the operating frequency based on faster units and allows more cycles for slower units. It achieves this by deploying a priority-based scheduling algorithm, where priority is given to faster execution units and if an instruction is assigned to a slower unit, additional cycles are assumed for execution to accommodate the longer cycle time of the unit. A recent work, ReViVaL [35], presents two fine-grained, post-fabrication tuning techniques. One technique offers an ability to add extra latency to pipelines that may exhibit longer-than-expected delays, and trades off latency to mitigate frequency degradation due to process variation. While effective, the extra latencies can overly degrade system-level performance (i.e., IPC) for latency-critical blocks with tight loops. In order to alleviate this compromise, ReViVaL proposes a fine-grained voltage-tuning technique called voltage interpolation. By allowing circuit blocks to statically choose between two supply voltages (VddH and VddL) after fabrication, different microarchitectural units within a processor can be viewed as operating at individually tuned "effective" voltage levels while maintaining a single consistent operating frequency (shown in Fig. 7.9). Voltage interpolation helps in combating the effect of within-die variations by providing a spatial dithering of voltages; speeding up slower paths and slowing down faster ones.

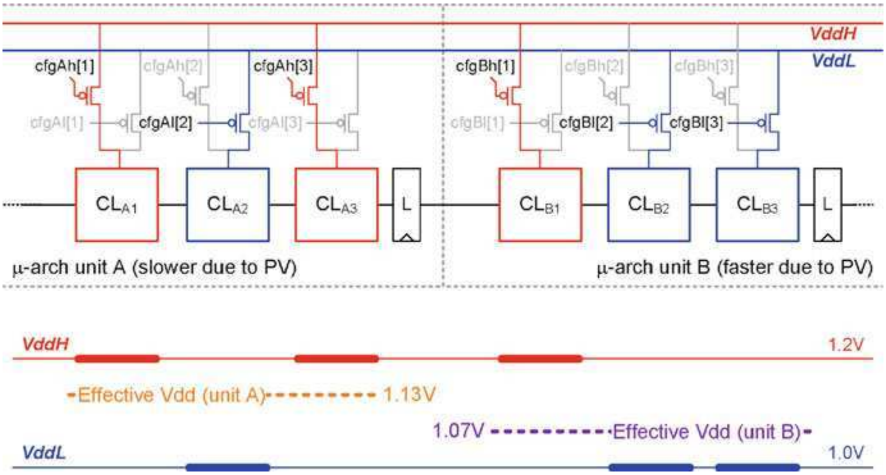


Fig. 7.9 Illustration of the voltage interpolation technique of ReViVaL. Two voltage supplies are assumed, and the ability to chose between two voltage provides fine-grained delay-tuning of a unit

7.4.2 Adapting to Thermal Variations

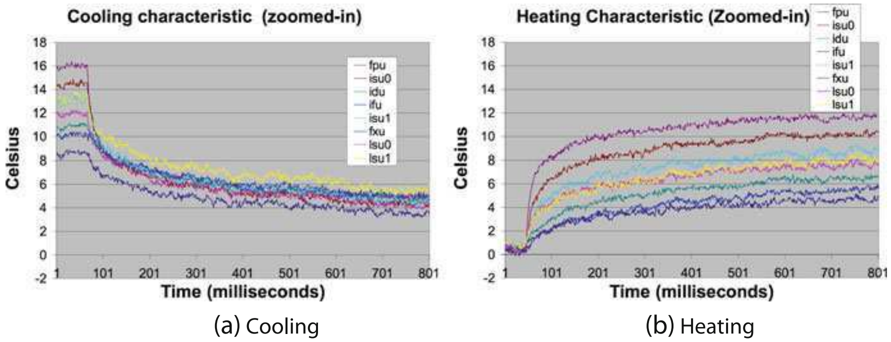
With technology scaling, the power densities across a chip are rising, leading to an overall increase of the processor temperature and also prominence of localized hots pots. Increase in the operating temperature across the chip can lead to worsening of both soft errors and aging-related errors, as these increase exponentially with temperature. Moreover, temperature also effects the speed of circuits, and circuits become slower at higher temperatures. Hence, to ensure performance and robustness guarantees the operating temperature cannot be allowed to rise beyond a certain range. Designers typically handle thermal variations by over-designing the thermal package (heat sink, fan, etc.) for the most severe hot spot that could arise, leading to an expensive solution. Architecture techniques can play an important role by allowing the package to be designed for the power dissipation of a typical application rather than a worst-case application [6, 56]. Such techniques enable dynamic thermal management (DTM) to adaptively control temperatures. DTM techniques rely on on-chip circuity that translates any signal of thermal stress into actuating a preventive action. These DTM techniques deploy global clock gating, fetch toggling, dynamic frequency scaling, or resource duplication to either prevent heating or relieve overheated resources in a super-scalar processor. Another solution has looked at the potential of managing lateral heat, which depends of on the functional unit adjacency. Sankaranarayanan et al. [49] propose a thermal-aware floorplanning for reducing peak processor temperature, and aim to even out the temperatures of the functional units through better spreading. A recent work by Paul et al. [41] proposes a memory-based computational framework that allows on-demand transfer of activity from functional units (integer ALU and floating point unit) to the embedded memory of the processor to reduce the thermal stress of a functional unit.

Moore's law and nanoscale technology are projecting the industry's move toward multiple simple cores on a single chip, leading to a "many-core" era of computing. SMT CMPs pose unique challenges and opportunities for power density, as they increase the throughput and thus on-chip heat. A recent work proposes a *heat-and-run* thread migration to alleviate the problem by migrating threads away from overheated cores to free SMT contexts on alternate cores [44].

7.4.2.1 Temperature-Aware Scheduling

In order to meet the challenge of mitigating thermal hot spots, several researchers have proposed techniques for making OS task schedulers thermally aware. For example, Choi et al. [10] describe several temperature-aware task scheduling algorithms that achieve significant reduction in temperature, with negligible performance loss. Reduction in temperature of several degrees in the current context of late CMOS era designs usually implies a sizeable reduction in power consumption as well, in view of the temperature dependence of leakage power.

The fundamental justification for implementing the concept of dynamic activity migration to system software is the relatively large thermal time constants involved in the case of a packaged microprocessor chip, functioning within a larger system. Figure 7.10 shows the measured decrease (or increase) in chip temperature as a function of time, as observed in Choi's work [10]. The data shown is for one of the cores within an IBM POWER5$^{\text{TM}}$ microprocessor chip, running a real workload.



(a) Cooling

(b) Heating

Fig. 7.10 Cooling and heating thermal time constant characterization (from [10])

As observed from the data, the thermal time constant for this chip, under typical packaging/cooling solutions, is around 100 ms. Regardless of the region of the core (e.g., the fixed point unit (fxu), the floating point unit (fpu), etc.), when a steady-state workload execution phase is abruptly terminated, the temperature drops to less than two-thirds of the high temperature in just over 100 ms. Similarly, in executing a workload from the ambient, no-load temperature point, the temperature rises to two-thirds of the final steady-state temperature in just over 100 ms. Thus, given that operating system task scheduling time slices are typically no more than 10 ms (and often much less), incorporating the activity migration heuristics into the OS task scheduler is expected to achieve the objective of reducing the spatial thermal gradients on the chip quite well.

Figure 7.11 shows the reduction in peak temperature as recorded in a typical thermal-aware task scheduling experiment, as reported in detail by Choi et al. [10]. The data show that the peak temperature may be reduced quite significantly (up to

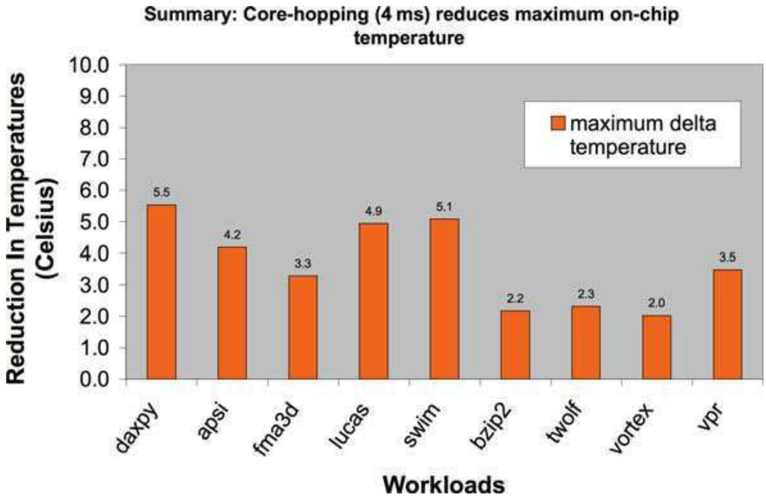


Fig. 7.11 Chip temperature reduction across various workloads (from [10])

5 or 6°C), by making the scheduler temperature-aware. As discussed in [10], this results in a significant reduction of the spatial temperature gradients on the chip, while saving considerable power, without giving up system performance.

7.4.3 Adapting to Voltage Variations

Large current swings over small time scales cause large voltage swings in the power-delivery subsystem due to parasitic inductance. The traditional way of dealing with voltage emergencies has been to over-design the system to accommodate worst-case voltage swings. A recent paper analyzing supply noise in a Power6 processor [26] shows the need for operating margins greater than 20% of the nominal voltage. Conservative processor designs with large margins ensure robustness. However, conservative designs either lower the operating frequency or sacrifice power efficiency.

In recent years, researchers have proposed tightening noise margins by adding mechanisms to the hardware that guarantee correctness in the presence of voltage emergencies. Such mechanisms enable more aggressive voltage margins, but at the expense of some run-time performance penalty. Architecture- and circuit-level techniques either proactively reduce activity when an emergency appears imminent [14, 27, 42, 43], or they reactively restore a correct processor state after an emergency compromises execution correctness [17]. Another set of solutions have been proposed that can tolerate emergencies, and eliminate most recurring emergencies by exploiting patterns in emergency behavior of a running code [18, 47, 46].

Proactive mechanisms strive to detect and avoid impending voltage emergencies arising from inductive noise to prevent failures [14, 27, 42, 43]. These impending voltage emergencies are either detected using voltage sensors [27] or current sensors [42]. Joseph et al. [27] propose a voltage sensor-based approach, where a throttling mechanism is invoked when the sensed supply voltage crosses a specified level, called the *soft threshold*. Such a throttling mechanism must react before the voltage deviation proceeds beyond the soft threshold to the hard threshold. The choice of the soft threshold level is largely governed by the voltage sensor response time and accuracy. The behavior of the feedback loop is determined by two parameters: the setting of the soft threshold level and the delays around the feedback loop.

There are several different ways to reduce current variations via throttling mechanisms such as frequency throttling, pipeline freezing, pipeline firing, issue ramping, or changing the number of the available memory ports [14, 27, 42, 43]. All of these mechanisms rely on voltage or current sensors to detect threshold crossings indicating timing-margin violations having occurred or about to occur. As shown in Fig. 7.12, the sensor turns the throttling mechanism (i.e., the actuator) on. Assuming that the control logic and actuation mechanism can react quickly, the main bottleneck in throttling for emergency-avoidance is the speed of the individual sensors and aggregation across multiple sensors in different parts of the processor. This delay sets the speed of the overall feedback loop. There are many ways to build

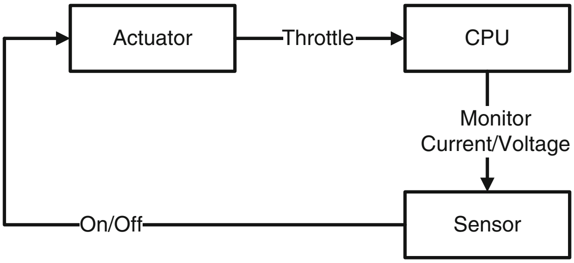


Fig. 7.12 Feedback loop in emergency-avoidance mechanisms: this figure highlights the feedback loop associated with proactive schemes. The main bottleneck being the delay associated with sensing and actuation mechanism

sensors with tradeoffs between delay and precision. Hence, it is important to understand the impact of the inherent delay and inaccuracy associated with the sensors on the different emergency-avoidance schemes.

Rather than trying to avoid voltage emergencies, a delayed commit and rollback (DeCoR [17]) mechanism does not attempt to avoid voltage emergencies altogether, but relies on a hardware mechanism that allows voltage emergencies to occur, but when they do, the architecture has a built-in mechanism to recover processor state. DeCoR leverages existing store queue and reorder buffers in modern processors to delay commit just long enough to verify whether an emergency has occurred. Performing a *rollback* in the event of an emergency [17] is akin to flushing the pipeline after a branch misprediction.

Another set of solutions [16, 18, 47, 46] have tried to exploit the repetitive behavior of voltage-noise induced timing-violations for designing solutions which can predict the occurrence of impending violations and hence trigger appropriate preventive measures. The system is implemented both in hardware and in software; the software component is at the level of a platform hypervisor. Figure 7.13 illustrates an overview of the system. The system relies on a *Detector* (hardware) that triggers a *Restarter* (hardware) to rollback execution whenever it detects an emergency [17]. To be able to predict emergencies and take preventive measures to eliminate them, it is necessary to find indicators based on activity leading to voltage noise.

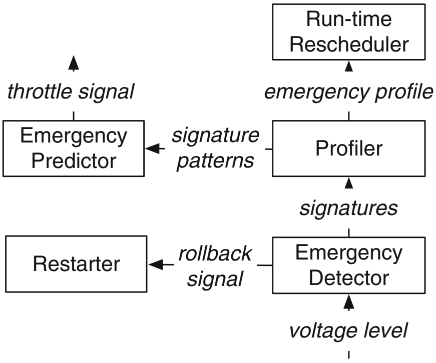


Fig. 7.13 Hardware–software collaborative system for handling voltage variations

Gupta et al. [18] show that there is a strong correlation between voltage violations and microarchitectural events. Reddi et al. [47] further augment isolated microarchitectural events with control flow sequence before and after a violation, called *emergency signatures*. Such signatures predict emergencies with over 90% accuracy [47]. The detector then feeds an emergency *Profiler* (software) with a signature that represents processor activity leading up to that emergency. The profiler accumulates signatures to guide a code *Rescheduler* (software) that eliminates recurrences of that emergency via run-time code transformations. The rescheduler, based on the control flow and microarchitectural event information extracted from the signature, performs code motion optimizations to smooth the original program activity. If code reshaping is unsuccessful, the *Profiler* arms an emergency *Predictor* (hardware) with signature patterns to instead predict and suppress recurrences of the emergency by throttling processor activity.

7.4.4 Dealing with PVT Variations

Almost all proposed solutions for dealing with parameter variations to date have focused on a single source of parameter variations at a time: temperature [56], voltage [27, 43], or process [35, 50, 65]. Implicitly, many of these studies assume that the costs of individual sources of parameter variations and the benefits of the proposed schemes are orthogonal. However, there are many complex interactions between parameter variations that highly depend on the underlying microarchitecture, workloads, and operating conditions. Figure 7.14 highlights the conservatives of simply adding up the margins from each source of variation. The figure shows the resulting timing margins from simulating all sources of variations together. When PVT variation effects are combined (PVT-combined), the average margin required reduces, but the spread between maximum and minimum margins increases. This

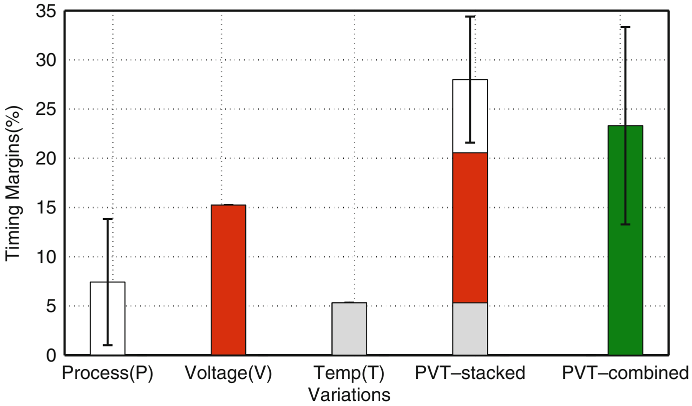


Fig. 7.14 Impact of PVT variations on timing margins: simple *stacking* leads to a larger mean and smaller spread in required timing margins as compared to the *combined* effects of all three

larger spread in margins primarily results from the interaction between voltage and process variations. Faster chips, consisting of transistors with lower threshold voltages, are less sensitive to voltage droops (sudden drops in voltage) and can operate with tighter margins. On the other hand, the transistors with higher threshold voltages in slower chips are more sensitive to voltage droop and require larger margins. Hence, the spread between maximum and minimum margins increases. The slowest chip, with the highest threshold voltages across the chip, exhibits lower leakage power to ameliorate thermal effects and slightly reduce the maximum required timing margin. Given that simply stacking margins misses important interactions found by considering PVT variations together, designers must also devise solutions that address all sources of variations in combination and not as individual, orthogonal components.

Sections 7.4.1, 7.4.2, 7.4.3 dealt with solutions to deal with each source of variation orthogonally. In this section we present the different solutions proposed to deal with combinations of these three sources of variations. These variations differ significantly in temporal and spatial scales. Process variations are static in nature, while voltage and temperature variations are highly sensitive to workload behavior, albeit at very different time scales. All sources of variation affect different parts of a microprocessor die in myriad ways with complex interactions. Microarchitectural techniques designed to mitigate parameter variations must clearly account for these differing characteristics.

Researchers have started exploring the opportunities of dynamically adapting processor parameters by exploiting workload variability. Tschanz et al. [67] explore schemes to dynamically adapt various combinations of frequency, Vdd and body bias to changes in temperature, supply noises, and transistor aging, to maximize average performance or improve energy efficiency. A prototype containing a TCP offload accelerator core is implemented. EVAL [50] presents a high-dimensional dynamic adaptation technique using a machine learning algorithm for maximizing performance and minimizing power in the presence of parameter variations. The adaptation mechanism mainly addresses process+temperature variations, and does not account for high-frequency voltage changes. Hence, the adaptation scheme is at a coarse-granularity of 120 ms.

A recent work proposes a framework called Tribeca for dealing with all the three sources of variation [19]. Tribeca illustrates the spatial and temporal characteristics of parameter variations by combining best-known modeling frameworks for P, V, and T with a POWER6-like microprocessor performance simulator augmented with circuit-extracted power models. Tribeca proposes a local recovery mechanism, which is essentially a per-unit recovery mechanism, that provides low-cost detection and rollback for timing-margin violations. This local recovery mechanism exploits spatial variation among units within the processor, and low cost recovery allows aggressive setting of design margins. Figure 7.15 illustrates the architectural support required for supporting a local-recovery mechanism for each unit. The fully distributed local recovery mechanism eliminates the need for a global recovery unit, distributing much of its functionality to the local recovery mechanisms.

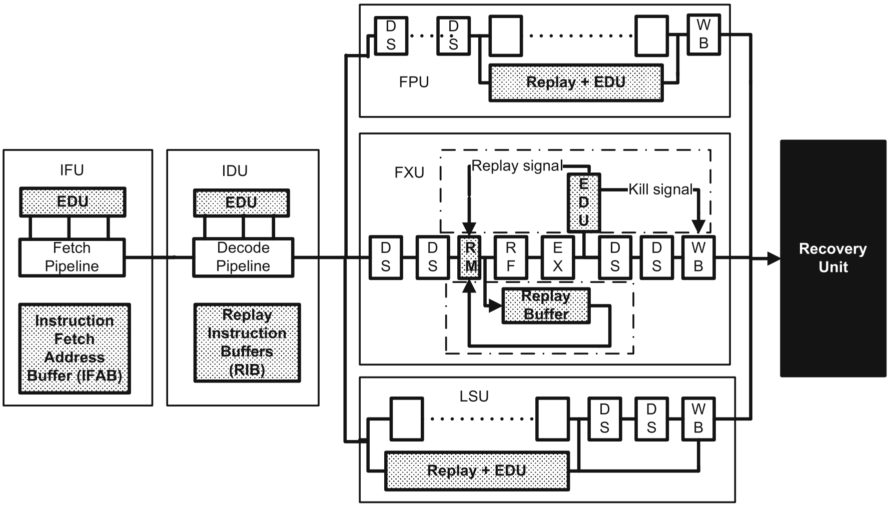


Fig. 7.15 Architectural support for local recovery mechanism: this figure shows the baseline microarchitecture with architectural support for local recovery mechanism. Additional hardware is shown by the shaded boxes. Details of the recovery mechanism for the execution units are presented for the fixed-point unit (FXU), and remaining units have the same logic represented as *Replay* + *EDU* logic. (DS, delay stages; RF, register file access; EX, execute; EDU, Error detection unit; RM, replay mux.). The recovery unit is replaced by the proposed distributed recovery mechanism

The smaller time constants associated with voltage variations coupled with the observation that voltage noise is heavily dependent on application characteristics [47] imply that solutions with finer temporal resolution will be necessary to target the combined effect of PVT-variations. Tribeca further explores fine-grained dynamic adaptation of processor frequency and voltage to exploit both temporal and spatial variation in delay across the processor. The adaptation mechanisms seek to maximize power-performance efficiency in the presence of parameter variations. The results show, under parameter variations, solutions that combine local recovery in tandem with a fine-resolution dynamic adaptation mechanism that can maximize performance with minimal increase in power.

7.4.5 *Tolerance to Aging-Induced Variations and Degradation in Circuit Timing*

As discussed before, the aging-induced variations generally cause a gradual degradation in the speed of propagation of signals within circuit blocks. Since different paths within a macro may age at different rates, depending on local process variability and actual activity profiles, the end result, in terms of circuit failure, is not very predictable. Initial failures may manifest as intermittent errors, for particular data patterns; but eventually, with time, the circuit block (and therefore the whole processor) may experience a catastrophic failure, if a lot of critical timing paths start to fail. In fact, the early warning of intermittent timing failures may not even manifest, depending on what the critical path distribution looks like for the given chip design.

The classical protection mechanism for aging-induced timing path variability is of course the well-known "design for worst case" situation. In this case, that would mean, adding in a timing "guard band" that factors in the worst-case additional delay degradation of a circuit block over a reasonable, expected lifetime of the product. As aging-related delay degradation characteristics increase in rate and spread, such a worst-case design solution would result in unacceptably large system performance overhead and may not even justify scaling to future technology nodes.

Many (if not all) of the broader system architecture-level protection paradigms that are "better than worst-case design" when it comes to aging-induced variations are arguably the same ones that also apply to other sources of variations that have already been discussed. For example, the DIVA [2] and Razor [12] paradigms (or small variants thereof) can apply generically as the foundation for a variation-tolerant design that would cover a large class of the sources of variation. We will not discuss these high-level error-tolerant architectural paradigms in this section; the reader is referred back to the introductory text at the beginning of Section 7.4 for that particular discussion.

We now discuss a few specific architecture-level solution methodologies to achieve target levels of mean time to failure, in spite of increasing failure rates attributable to aging-induced variations and attendant circuit degradation.

7.4.5.1 Lifetime Reliability-Aware Microarchitectures

Srinivasan et al. [62] first proposed architecture-level solution approaches to combat the problem of aging-induced failure mechanisms like electromigration, oxide breakdown, thermal cycling, and NBTI. Using the microarchitecture-level lifetime reliability model called RAMP (see Section 7.3), a couple of different lifetime extension techniques were explored. One set of techniques lets designers apply selective redundancy at the structure level and/or exploit existing microarchitectural redundancy [61]. Another technique is dynamic reliability management (DRM) [59].

In employing structural redundancy to enhance system reliability, Srinivasan et al. [61] have attempted to study the optimal tradeoffs across lifetime reliability, performance, and cost. The added redundant structures would allow run-time reconfiguration resulting in longer processor lifetimes. Specifically, three techniques were examined to explore opportunities of efficient reliability enhancement:

(a) Structural Duplication (SD): In SD, extra structural redundancy is added to augment the required base processor resources during microarchitectural specification. The extra structures that are added are designated as spares, and are power gated (i.e., not used) at the beginning of a processor's lifetime. During the course of the processor's life, if a structure with an available spare fails or is determined to be on the verge of failure, the processor would reconfigure and use the spare structure. This extends the processor's life beyond the point when it would have normally failed. Instead, processor failure would occur only when a structure without a spare fails, or when all available spares for a structure fail. SD increases processor reliability without any loss of performance. However, duplication of resources adds a cost (due to increased die area) overhead to the original microarchitecture area. In the work by Srinivasan et al. [61], SD was used to augment a single processor core with spare resources at the functional unit level (e.g., integer unit or floating point unit). The concept applies equally well, and perhaps with greater practical relevance to the case of adding spare cores, memory controllers and global storage or communication resources in the context of modern multi-core processor chips.

(b) Graceful Performance Degradation (GPD): GPD allows existing processor redundancy to be leveraged for lifetime enhancement without the addition of extra (spare) resources. Most modern high-performance microprocessors already use redundancy to exploit available parallelism in common applications. However, only a subset of these resources are required for functional correctness, if the chip microarchitecture is able to deal with failed resources. For example, a processor core (e.g., the one in the IBM POWER5 multi-core chip [55]) may have two of each execution resource, e.g., floating point unit, integer (fixed point) arithmetic unit, load-store unit, etc. If the core microarchitecture were architected in such a way that it would remain functional (at reduced performance) even if one of any type of execution units became faulty due to aging-induced degradation, then one would have a processor with a GPD feature. Similarly, at a higher level of functional abstraction, if one of the cores of a multi-core chip failed over time due to aging, a GPD-architected design would enable the system to remain functional, albeit at reduced multiprocessor performance.

Figure 7.16 shows a summary result from Srinivasan et al. [61] that shows the effect of SD, GPD, and combinations, applied at the processor core level. The processor core assumed was a POWER4/5 class core, without multi-threading. The normalized performance/cost (or P/C) for all the applications on the baseline processor design point is 1.0. In SD, the cost will increase, leading to P/C values lower than 1.0. In GPD, performance will decrease, leading to P/C values lower than 1.0, and in SD+GPD, both increases in cost and decreases in performance lower the value of P/C. Figure 7.16 shows the average MTTF benefit across all applications analyzed in this research from each of the three techniques (SD, GPD, and SD+GPD) for a range of P/C values. The vertical axis represents normalized MTTF. The horizontal axis represents different P/C design points. For both GPD and SD+GPD, guaranteed as well as actual performance levels are evaluated. At high P/C values (i.e., low overhead in terms of cost or performance), GPD provides much more benefit than SD. However, the benefit from GPD tapers off as we move to lower values of P/C. On the other hand, SD provides much more MTTF benefit at lower P/C

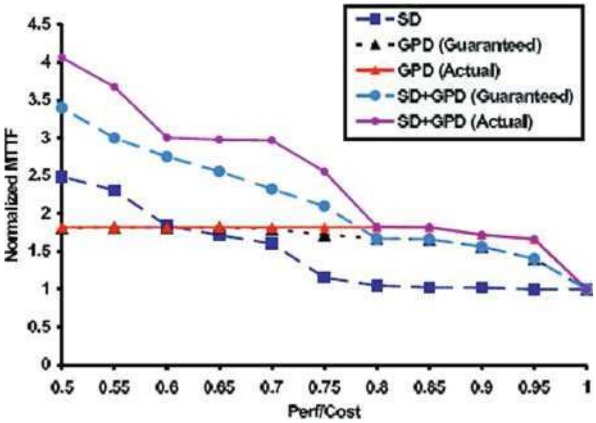


Fig. 7.16 Average normalized mean-time-to-failure (MTTF) benefit versus performance/cost for SD, GPD, and SD+GPD across all applications. For GPD and SD+GPD, both guarantees and actual performance values are given

values, and overtakes GPD. The combination of both techniques always provides the highest MTTF benefit. This is intuitive because SD+GPD can choose any configuration that SD or GPD could choose, in addition to the cross product of the two. However, SD+GPD chooses the same configurations as GPD chooses at high values of P/C. Finally, since the processors run at full performance at the beginning of their lifetime, the same MTTF benefit for GPD (Actual) and SD+GPD (Actual) comes at higher P/C values than GPD (Guaranteed) and SD+GPD (Guaranteed).

In Dynamic Redundancy Management (DRM), the processor uses run-time adaptation to respond to changing application behavior in order to maintain its lifetime reliability target. In contrast to worst-case design practices, DRM lets manufacturers qualify reliability at lower (but more likely) operating points than the worst case. As in dynamic thermal management [6], if applications exceed the failure rate design limit, the processor can adapt by throttling performance to maintain the system reliability target. Conversely, for applications that do not stress the reliability limit, DRM uses adaptation to increase performance while maintaining the target reliability. As explained by Srinivasan et al. in [62], DRM offers the potential for cost benefits without performance loss. A worst-case design principle, in this context, may (for example) require qualifying the design to operate without failure at a rather high chip temperature of $T_{\text{qual}} = 400K$. Changing the reliability design point to $T_{\text{qual}} = 370K$ would save design cost without slowing any of the applications considered in the RAMP-based analysis study [62, 59]—which shows that for this class of aging-induced variation (leading to failures), worst-case reliability qualification is unnecessarily conservative. Using DRM, and allowing a small performance degradation, a design team can further lower the reliability qualification cost. In the results reported by Srinivasan et al. [59, 62], even a T_{qual} of 345°K, performance loss was limited.

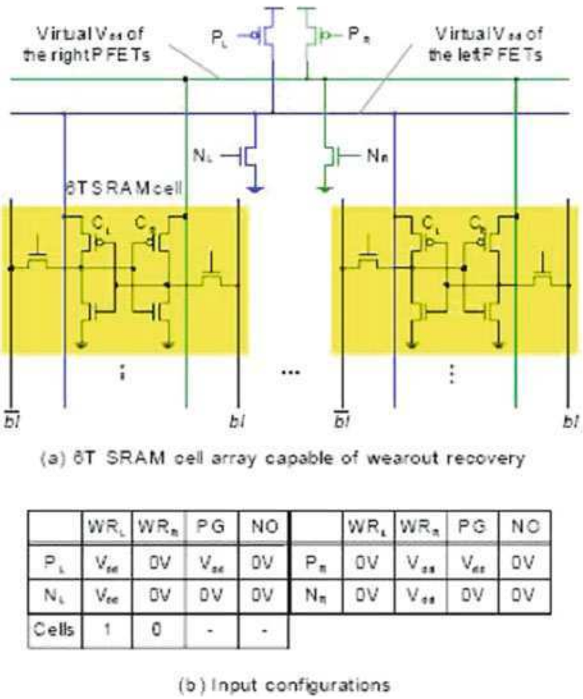
7.4.5.2 Proactive Redundancy or Wear-Leveling

Shin et al. [54] first advanced the architectural concept of proactive redundancy, in which an architected spare is put into active use proactively, even before an actual failure in one of the regular resources has occurred. The technique proposed in this work exploits the principle of wear-leveling, which has seen more recent application in domains like solid-state disk microcontrollers [8] and phase change memory systems [45]. The basic concept is to enforce an even wearout profile across the resources of interest. For ease of illustration, let us consider an n-core microprocessor chip, with one extra core that serves as a spare. Thus, we have $(n + 1)$-core system in which one core is always inactive (as a spare). If the spare core is a fixed, specially designated physical core that is brought into action only when one of the n active cores encounters a failure, then the n-way system fails on encountering a second core failure. In such a system, the time to the first failure is determined by the "weakest" of the n active cores. Due to process variation, there will always be one core that has the tendency to wear out most rapidly. Furthermore, even if the cores are virtually identical in terms of wearout characteristics, workload variability might tend to make one core wear out the fastest. If there is no spare core in the system, the wearout across the active cores could be made more uniform, by making the task scheduler smarter, as proposed by Kursun and Cher [32]. If there is a spare core, a hardware—software mechanism that allows "spare rotation" even without the onset of the first failure can be shown to extend the lifetime of the system significantly over the reactive replacement method. This is especially true, if the particular failure mechanism has the characteristics of partial reversal of aging through relaxation (as in the case of electromigration and NBTI).

We now present an example from the prior work of Shin et al. [54] to explain the benefit of the proactive redundancy concept in quantitative terms. Let us focus on a particular failure mechanism: NBTI. In the case of NBTI, the effects of wearout can be reversed during the suspended period that stress conditions are removed (i.e., during recovery mode) [54, 70]. Thus, proactive use of even a limited amount of redundancy can suspend or reverse overall component wearout quite significantly. Reactive use of redundancy provides no such benefits to component wearout but, instead, provides only for as many wearout failures to be tolerated as there are redundant components, which typically is very limited. Furthermore, with proper scheduling, proactive use of redundancy allows component-level wear-leveling, and this in itself prevents pre-mature chip failure caused by one part or component of the chip to age quickly (e.g., due to workload imbalance).

Recovery from NBTI-induced threshold voltage shift can occur during the period over which no stress is applied on the gate (i.e., $V_{\mathrm{gs}} = 0$) as hydrogen atoms diffused during NBTI stress return to the interface to mend the dangling bonds and electrons injected from the substrate neutralize the oxide traps created from NBTI stress [54, 70]. This naturally occurring recovery effect of NBTI-induced wearout is intensified (i.e., made faster and more pronounced) when PFET devices are reverse biased (i.e., $V_{\mathrm{gs}} = V_{\mathrm{dd}}$) as hydrogen atoms are more effectively attracted to the interface and electron injection is more active [71]. An SRAM array design that allows PFET

Fig. 7.17 Circuit-level SRAM array design which allows NBTI wearout recovery of 6T memory cells by reverse biasing the PFET devices in the cells. Input combinations NO, PG, WR_L, and WR_R are those needed for normal (active), power gating, and wearout recovery (of the left and right PFETs, respectively) modes of operation



(a) 6T SRAM cell array capable of wearout recovery

	WR_L	WR_R	PG	NO		WR_L	WR_R	PG	NO
P_L	V_{dd}	0V	V_{dd}	0V	P_R	0V	V_{dd}	V_{dd}	0V
N_L	V_{dd}	0V	0V	0V	N_R	0V	V_{dd}	0V	0V
Cells	1	0	-	-					

(b) Input configurations

devices to undergo intensified NBTI wearout recovery is proposed by Shin et al. [54] and is shown again in Fig. 7.17 below.

Conventional power reduction techniques like voltage scaling reduce NBTI stress conditions to a certain degree by reducing the applied electric field. This results in a slowdown of wearout. Power gating, if properly architected could eliminate the electric field stress, and this facilitates a limited degree of recovery. However, even with power gating, complete elimination of the electric field may not be possible. In order to accelerate recovery and repair the delay degradation to the fullest extent possible, a 6T SRAM cell array design is proposed in Shin [54], as depicted in Fig. 7.17.

The PFET device must be subject to reverse bias. Figure 7.17, shows a 6T SRAM cell array design that is capable of fully exploiting NBTI intensive recovery effects when operating in wearout recovery mode while maintaining normal operation in active mode. As shown in the figure, reverse biasing is created by charging the gate of the PFET devices and discharging the sources of the devices. Since each SRAM cell has cross-coupled inverters, each inverter is used to charge the gate of the PFET device of the other by storing the proper value to the cell before transitioning to recovery mode. That is, a “1” needs to be written for recovery of the left PFET devices of the cells (C_L) while a “0” needs to be written for recovery of the right PFET devices of the cells (C_R).

Also, the sources of the left and right PFETs need to be tied to separate virtual V_{dd} rails in order to provide the proper voltage level. The virtual V_{dd} of PFETs under recovery mode is discharged through the NFET device (N_L or N_R and that of the other side is charged through the PFET device (P_L or P_R). The regular structure of SRAM arrays enables the separation of virtual V_{dd} for the left and right PFETs of the cells to be relatively straightforward. The power line running vertically across array cells to supply V_{dd} to both sides of the PFETs of cells can be divided into two, one for each side, each connected to different virtual power rails as illustrated in Fig. 7.17a. Since most SRAM array designs already have virtual power or ground rails for cell biasing, this implementation of wearout recovery mode costs negligibly small additional area overhead. Figure 7.17b depicts the SRAM array configurations for active (normal) power gating and wearout recovery modes of operation of the left and right PFETs of the cells.

Figure 7.18 shows an 8-way set-associative cache consisting of 64 arrays, eight of which compose one associative way (each row in the figure) and each of which is implemented as illustrated in Figure 7.17 to enable NBTI wearout recovery. One additional spare array is also implemented and used proactively to allow any one of the 65 arrays to operate in recovery mode at any given time.

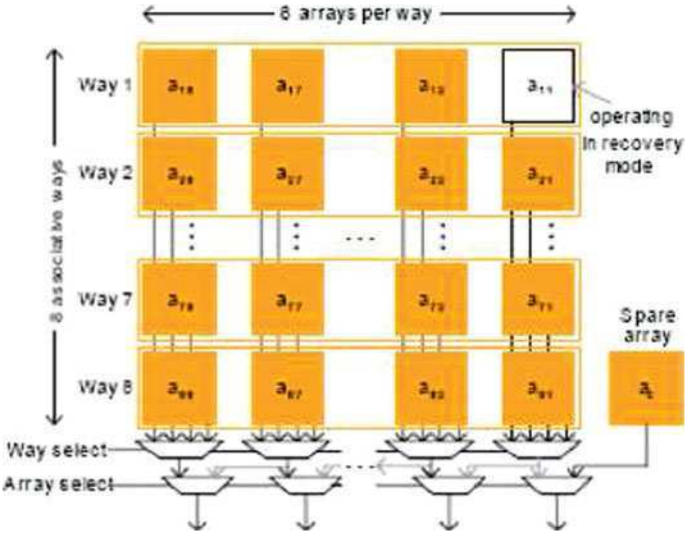


Fig. 7.18 Cache SRAM configured to support proactive use of array-level redundancy for wearout recovery

In the work by Shin et al. [54], the benefit analysis of proactive redundancy was studied in the context if an L2 cache within a POWER5-like processor chip. The cache SRAM structure is assumed to support the proactive redundancy approach described in the preceding paragraphs of this section. Various different heuristic policies were studied with the goal of extending the mean time to failure (MTTF) of the L2 cache. The analysis reported in [54] shows that with one redundant array used proactively for intensive wearout recovery in which 70% of the wear out can be recovered during the recovery mode, the lifetime reliability of the cache can be improved by a factor of 5.5–10.2 compared to the baseline 64-array case, without any redundancy. The improvement factor is 3–5 times over a conventional reactive redundancy technique, in which the spare array is phased into operation only after the first failure in the main array banks is detected. The performance loss incurred by the proactive redundancy policies is found to be very small [54].

References

1. Agarwal A, Blaauw D, Zolotov V (2003) Statistical timing analysis for intra-die process variations with spatial correlations. In: International conference on computer-aided design, San Jose, CA
2. Austin TM (1999) DIVA: a reliable substrate for deep submicron microarchitecture design. In: MICRO 32: proceedings of the 32nd annual ACM/IEEE international symposium on microarchitecture, Haifa, Israel
3. Aygun K, Hill MJ, Eilert K, Radhakrishnan R, Levin A (2005) Power delivery for high-performance microprocessors. Intel Technol J 9(4):26–44
4. Borkar S (2005) Designing reliable systems from unreliable components: the challenges of variability and degradation. IEEE Micro 25(6)
5. Bowman KA, Tschanz JW, Kim NS, Lee J, Wilkerson CB, Lu SL, Karnik T, De V (2008) Energy-efficient and metastability-immune timing-error detection and instruction replay-based recovery circuits for dynamic variation tolerance. In: ISSCC 2008
6. Brooks D, Martonosi M (2001) Dynamic thermal management for high-performance microprocessors. In: International symposium on high-performance computer architecture (HPCA)
7. Brooks D, Bose P, Schuster S, Jacobson H, Kudva P, Buyuktosunoglu A, Wellman JD, Zyuban V, Gupta M, Cook P (2000) Power-aware microarchitecture: design and modeling challenges for the next generation microprocessors. IEEE Micro 20(6)
8. Chang LP, Du CD (2009) Design and implementation of an efficient wear-leveling algorithm for solid-state-disk microcontrollers 15(1) ACM Transactions on Design Automation of Electronic:1–36
9. Chen TH, Chen CCP (2001) Efficient large-scale power grid analysis based on preconditioned krylov-subspace iterative methods. In: 38th conference on design automation, Las Vegas, NV
10. Choi J, Cher C, Franke H, Haman H, Weger A, Bose P (2007) Thermal-aware task scheduling at the system software level. In: International symposium on low power electronics and design (ISLPED) Portland, OR
11. El-Essawy W, Albonesi D (2004) Mitigating inductive noise in SMT processors. In: International symposium on low power electronics and design, Newport, CA
12. Ernst D, Kim N, Das S, Pant S, Rao R, Pham T, C Ziesler D Blaauw KF T Austin, Mudge T (2003) Razor: a low-power pipeline based on circuit-level timing speculation. In: MICRO'03, San Deigo, CA
13. Gara A, et al (2005) Overview of the blue gene/l system architecture. IBM J Res Technol 49(2/3):195–212
14. Grochowski E, Ayers D, Tiwari V (2002) Microarchitectural simulation and control of di/dt-induced power supply voltage variation. In: International symposium on high-performance computer architecture, Boston, MA
15. Gupta MS (2009) Variation-aware processor architectures with aggressive operating margins. PhD thesis, Harvard University, Cambridge, MA
16. Gupta MS, Rangan K, Smith MD, Wei GY, Brooks DM (2007) Towards a software approach to mitigate voltage emergencies. In: ISLPED '07, Portland, OR

17. Gupta MS, Rangan K, Smith MD, Wei GY, Brooks DM (2008) DeCoR: a delayed commit and rollback mechanism for handling inductive noise in processors. In: International symposium on high-performance computer architecture (HPCA), Salt Lake City, UT
18. Gupta MS, Reddi VJ, Holloway G, Wei GY, Brooks D (2009) An event-guided approach to handling inductive noise in processors. In: Design automation and test in Europe (DATE), Nice
19. Gupta MS, Rivers J, Bose P, Wei GY, Brooks D (2009) Tribeca: design for PVT variations with local recovery and fine-grained adaptation. In: International symposium on microarchitecture (MICRO), New York, NY
20. Herell D, Becker B (1999) Modelling of power distribution systems for high-performance microprocessors. In: IEEE transactions on advanced packaging, vol 22, pp 240–248
21. Hicks J, et al (2008) 45 nm transistor reliability. Intel Technol J 12(2):131–145
22. Hu CK, Gignac L, Rosenburg R (2006) Electromigration of cu/low dielectric constant interconnects. Microelectron reliab 46(2–4):213–231
23. Humenay E, Tarjan D, Skadron K (2007) Impact of systematic process variations on symmetrical performance in chip multiprocessors. In: Design, automation and test in Europe (DATE), Nice
24. Iyer A, Marculescu D (2002) Power and performance evaluation of globally asynchronous locally synchronous processors. In: International symposium on computer architecture, Anchorage, AK
25. Jackson K, Wisniewski M, Schmidt D, Hild U, Heisig S, Yeh P, Gellerich W (2009) IBM System z10 performance improvements with software and hardware synergy. IBM J Res Dev 53(16):1–8
26. James N, Restle P, Friedrich J, Huott B, McCredie B (2007) Comparison of split-versus connected-core supplies in the POWER6 microprocessor. In: ISSCC 2007
27. Joseph R, Brooks D, Martonosi M (2003) Control techniques to eliminate voltage emergencies in high performance processors. In: International symposium on high-performance computer architecture, Annaheim, CA
28. Kalla R, Sinharoy B, Starke W, Floyd M (2010) Power7™: IBM's next generation server processor. IEEE Micro 30(2):7–15
29. Krishnan AT, Reddy V, Aldrich D, Raval J, Christensen K, Rosal J, O'Brien C, Khamankar R, Marshall A, Loh WK, McKee R, Krishnan S (2006) Sram cell static noise margin and vmin sensitivity to transistor degradation. In: International electron devices meeting
30. Krum A (2000) Thermal management. In: Kreith F (ed) The CRC handbook of thermal engineering, CRC, Boca Raton, FL
31. Kurd N, Douglas J, Mosalikanti P, Kumar R (2008) Next generation intel micro-architecture (Nehalem) clocking architecture. IEEE symposium on VLSI circuits
32. Kursun E, Cher CY (2009) Variation-aware thermal characterization and management of multi-core architectures. In: International conference on computer design (ICCD), Lake Tahoe, CA
33. Li Y, Kim YM, Mintarno E, Mitra S, Gardner DS (2009) Overcoming early-life failure and aging for robust systems. IEEE Design and Test 26(6):28–39
34. Liang X, Brooks D (2006) Mitigating the impact of process variations on processor register files and execution Units. In: International symposium on microarchitecture (ISCA), Boston, MA
35. Liang X, Wei GY, Brooks D (2008) ReVIVaL: Variation tolerant architecture using voltage interpolation and variable latency. In: International symposium on computer architecture (ISCA-35), Beijing, China
36. McPherson J et al (2003) Trends in the ultimate breakdown strength of high dielectric-constant materials. IEEE Trans Electron Devices 50(8):1771–1778
37. Mukherjee S (2008) Architecture design for soft errors. Morgan Kaufmann, San Francisco, CA

38. Ndai P, Bhunia S, Agarwal A, Roy K (2008) Within-die variation aware scheduling in superscalar processors for improved througput. IEEE Trans Compu 57: 940–951
39. Pae S, Maiz J, Prasad C, Woolery B (2008) Effect of bti degradation on transistor variability in advanced semiconductor technologies. IEEE Trans Device Mater Reliab 8(3):519–525
40. Park D, Nicopoulos C, Kim J, Vijaykrishnan N, Das CR (2006) Exploring fault-tolerant network-on-chip architectures. In: Proceedings of international conference on dependable systems and networks (DSN), Philadelphia, PA, pp 93–104
41. Paul S, Bhunia S (2010) Dynamic transfer of computation to processor cache for yield and reliability improvement. IEEE transactions on VLSI
42. Powell M, Vijaykumar TN (2004) Exploiting resonant behavior to reduce inductive noise. In: ISCA, Munchen
43. Powell MD, Vijaykumar TN (2003) Pipeline muffling and a priori current ramping: architectural techniques to reduce high-frequency inductive noise. In: Int'l symposium on low power electronics and design, Seoul
44. Powell MD, Gomaa M, Vijaykumar T (2004) Heat-and-run: leveraging SMT and CMP to manage power density through the operating system. In: International conference on architectural support for programming languages and operting systems (ASPLOS), Boston, MA
45. Qureshi MK, Karidis J, Franceschini M, Srinivasan V, Lastras L, Abali B (2009) Enhancing lifetime and security of phase change memories via start-gap wear leveling. In: International symposium on microarchitecture (MICRO), New York, NY
46. Reddi VJ, Campanoni S, Gupta MS, Smith MD, Wei GY, Brooks D (2009) Software-assisted hardware reliability: abstracting circuit-level challenges to the software stack. In: Design automation conference (DAC), San Francisco, CA
47. Reddi VJ, Gupta MS, Holloway G, Smith MD, Wei GY, Brooks D (2009) Voltage emergency prediction: using signatures to reduce operating margins. In: International symposium on high-performance computer architecture (HPCA), Raleigh, NC
48. Sanda PN, Kellington JW, Kudva P, Kalla R, McBeth RB, Ackaret J, Lockwood R, Schumann J, Jones CR (2008) Soft-error resilience of the IBM POWER6 processor, IBM J Res Develop 52(3):221–314
49. Sankaranarayanan K, Velusamy S, Stan M, Skadron K (2005) A case for thermal-aware floorplanning at the microarchitectural level. J Instr-Level Parall 7
50. Sarangi S, Greskamp B, Tiwari A, Torrellas J (2008) EVAL: utilizing processors with variation-induced timing errors. In: 41st international symposium on microarchitecture (MICRO), Lake Como
51. Sarangi SR, Greskamp B, Teodorescu R, Nakano J, Tiwari A, Torrellas J (2008) VARIUS: a model of process variation and resulting timing errors for microarchitects. IEEE Trans on Semiconductor Manufacturing (IEEE TSM)
52. Semeraro G, Magklis G, Balasubramonian R, Albonesi DH, Dwarkadas S, , Scott ML (2002) Energy-efficient processor design using multiple clock domains with dynamic voltage and frequency scaling. In: International symposium on high-performance computer architecture (HPCA), Boston, MA
53. Shin J, Zyuban V, Hu Z, Rivers JA, Bose P (2007) A framework for architecture-level lifetime reliability modeling. In: DSN '07: proceedings of the 37th Annual IEEE/IFIP international conference on dependable systems and networks, Edinburgh
54. Shin J, Zyuban V, Bose P, Pinkston TM (2008) A proactive wearout recovery approach for exploiting microarchitectural redundancy to extend cache SRAM lifetime. In: International symposium on computer architecture (ISCA-35), Beijing, pp 353–362
55. Sinharoy B, Kalla R, Tendler J, Eickemeyer R, Joyner J (2005) POWER5 system microarchitecture. IBM J Res Technol 49(4/5):505–523
56. Skadron K, Stan MR, Huang W, Velusamy S, Sankaranarayanan K, Tarjan D (2003) Temperature-aware microarchitecture. In: International symposium on computer architecture (ISCA-30), San Deigo, CA

57. Smith L, Anderson RE, Roy T (2001) Chip-package resonance in core power supply structures for a high power microprocessor. In: Proceedings of the Pacific Rim/ASME international electronic packaging technical conference and exibition
58. Spainhower L et al (1992) Design for fault tolerance in ES 9000 Model 900. In: Proceedings of fault-tolerant computing symposium
59. Srinivasan J, Adve S, Bose P, Rivers J (2004) The case for lifetime reliability-aware microprocessors. In: International symposium on computer architecture (ISCA), Munchen
60. Srinivasan J, Adve SA, Bose P, Rivers J (2004) The impact of technology scaling on lifetime reliability. In: International conference on dependable systems and networks (DSN), Florence
61. Srinivasan J, Adve S, Bose P, Rivers J (2005) Exploiting structural duplication for lifetime reliability. In: International symposium on computer architecture (ISCA), Madison, WI
62. Srinivasan J, Adve S, Bose P, Rivers J (2005) Lifetime reliability: toward an architectural solution. In: IEEE Micro
63. Stathis JH (2002) Reliability limits for the gate insulator in cmos technology. IBM J Res Technol 46:265–286
64. Stathis JH, Zafar S (2006) The negative bias temperature instability in mos devices: a review. Microelectron Reliab 46(2–4):270–286
65. Teodorescu R, Nakano J, Tiwari A, Torrellas J (2007) Mitigating parameter variation with dynamic fine-grain body biasing. In: MICRO, Chicago, IL
66. Tschanz J, Kao J, Narendra S (2002) Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage. J Solid-State Circuits 37(11):1396–1402
67. Tschanz J, Kim NS, Dighe S, Howard J, Ruhl G, Vangal S, Narendra S, Hoskote Y, Wilson H, Lam C, Shuman M, Tokunaga C, Somasekhar D, Tang S, Finan D, Karnik T, Borkar N, Kurd N, De V (2007) Adaptive frequency and biasing techniques for tolerance to dynamic temperature-voltage variations and aging. In: ISSCC 2007
68. Wu E et al (2002) Interplay of voltage and temperature acceleration of oxide breakdown for ultra-thin gate dioxides, pp 1787–1798
69. ISSCC 2010 Trends Report: http://isscc.org/doc/2010/ISSCC2010_TechTrends.pdf
70. Zafar S (2005) Statistical mechanics based model fo negative bias temperature instability (nbti) in mosfets. J Appl Phys 97(10):103709
71. Zafar S et al (2005) Threshold voltage instabilities in high-k gate dielectric stacks. IEEE Trans Dev Mater Reliab 5(1):45–64

Chapter 8
Low-Power and Variation-Tolerant Application-Specific System Design

Georgios Karakonstantis and Kaushik Roy

Abstract This chapter presents an overview of several representative techniques that can potentially have large impact on the design of low-power and variation-tolerant application-specific systems. Common characteristic of the presented approaches is that they exploit properties of digital signal processing (DSP) algorithms and combine circuit and architecture level techniques in order to provide intelligent trade-offs between circuit level metrics such as power and performance with system level metrics such as quality-of-results and tolerance to variations (yield). The techniques presented in this chapter target various types of designs that include logic and memory architectures and complete DSP systems.

8.1 Introduction

Personal mobile communications – anytime, anywhere access to data and communication services – have been continuously increasing since the operation of the first cellular phone system. This growth is combined with increasing demand by consumers for intelligent, small, and multifunctional products able not only to transmit voice messages but also color pictures and video images, e-mails, as well as to provide access to the internet [23]. Such demand for richer services, multifunctional portable devices, and high data rates can be only envisioned due to the improvement in semiconductor technology. Technology scaling has enabled improvements in the three major design optimization objectives: increased performance, lower power consumption, and lower die cost, while system design has focused on bringing more functionality into products at lower cost. Unfortunately, this *happy scaling* scenario due to the barriers that are faced in sub-90 process nodes – increased power dissipation and parametric variations – may slow down or come to an end in the foreseeable future.

G. Karakonstantis (✉)
Department of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, USA
e-mail: gkarakon@purdue.edu

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_8,

As operating frequency and integration density increase, power dissipation increases. This is evident from the fact that due to the demand for increased functionality on a single chip, more and more transistors are being packed on a single die and hence, the switching frequency increases every technology generation. Switching power dissipation due to dynamic and short circuits components are progressively increasing since more number of transistors switch at any particular time instant [50]. In addition estimated improvements in capacity of batteries are much slower than what is needed to support current and future mobile applications [48]. Therefore the limited energy available in the next-generation tiny and multifunctional portable devices must be efficiently used. This has made the design of low-power and high-performance digital signal processing (DSP) applications, which are the heart of today's and future portable devices, more important than ever [14, 47].

However, the design of low-power and high-performance circuits is further complicated due to increased process parameter variations in nanometer regime [8, 55] as discussed in previous chapters. *Inter-die* and *intra-die* [10] variations have unpredictable impact on the performance of the nano-scaled devices since they result in fluctuations in transistor length (L), width (W), flat-band voltage (V_{FB}), and oxide thickness (T_{ox}) and lead to a large spread in the threshold voltage (V_{th}) of transistors. The effect of the V_{th} variation on the delay is evident from the alpha power model [51] according to which the gate delay can be approximated by

$$D_{\mathrm{G}} = \frac{\mathrm{K}V_{\mathrm{dd}}}{(V_{\mathrm{dd}} - V_{\mathrm{th}})^{a}} \tag{8.1}$$

Any V_{th} variation indicates that a circuit designed to meet a target delay using nominal V_{th} transistors may not meet the delay target, leading to delay failures. Such delay failures can potentially result in incomplete computations, leading to complete failure or degrading significantly the output quality of the application under consideration. Hence, parametric yield of a circuit (probability to meet the desired performance or power specification) is expected to suffer significantly. Furthermore, V_{th} variation poses concern in the operation of Static Random Access Memory (SRAM) [6], a ubiquitous component in embedded systems. Therefore it is crucial to effectively deal with parametric variations to maintain adequate yield/quality and to ensure correct operation of future systems in addition to ensuring low-power operation.

8.1.1 Addressing Contradictory Design Requirements

Several techniques at various levels of hardware stack, circuit, and architecture, have been developed in order to address low-power and parameter variations. Such techniques can be broadly classified into design-time and run-time where parameter shift is detected and adjusted after manufacturing by changing operating parameters such as supply voltage, frequency, or body bias [48].

8.1.1.1 Power Consumption Reduction

It is well known that the power dissipation of a circuit consists of two components, the dynamic (P_{dyn}) and static (P_{stat}) power [50]:

$$P = P_{\text{dyn}} + P_{\text{stat}} = C_{\text{eff}}\, {V_{\text{dd}}}^2 f + V_{\text{dd}}(I_{\text{sub}} + I_{\text{gate}}) \tag{8.2}$$

where C_{eff} is the effective switching capacitance of the circuit, f is the frequency of operation, and I_{sub} is the subthreshold current, whereas I_{gate} is the gate direct tunneling current. Given such relationship, several techniques have been proposed that either try to reduce the switching activity [14], or the frequency or the supply voltage of the circuit in order to reduce the power consumption. However, the most effective method for power minimization is voltage scaling or over-scaling (VOS) since it leads to large improvements in power consumption due to quadratic dependence of power on supply voltage [48]. VOS extends the concept of voltage scaling beyond the critical voltage value at which the critical path delay (clock period) imposed by architecture, and application is just met (Fig. 8.1). Scaling beyond such critical voltage is referred to as VOS and can be considered as another source of noise in integrated circuits [27, 54]. Specifically, VOS increases the delays in all computation paths and can result in incorrect computation of certain operations, leading to drastic degradation of output quality in the application layer. Therefore, applying supply voltage scaling randomly to any design can adversely affect the output, leading to lower manufacturing/parametric yield [24]. Therefore, design methodologies are required that help maintain the same frequency of operation with minimal penalty on throughput under VOS.

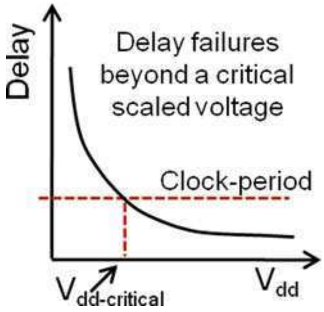


Fig. 8.1 Voltage over-scaling

8.1.1.2 Tolerance to Parameter Variations

The sub-90 nm era has witnessed a surge in research efforts directed at modeling and addressing the effects of variations. The majority of the traditional logic and circuit level approaches, such as adaptive body biasing (ABB), adaptive supply voltage scaling (ASVS), and clock tuning [7, 17, 35, 57, 58] aim at providing proper timing margins, while assuming worst-case process conditions. Specifically, at design time, worst-case conditions and their effect on performance of various parts of the circuit are estimated. Based on such estimations transistors are up-sized or voltage is up-scaled to ensure adequate timing margins that subsequently guarantee correct operation under worst-case parametric variations [58]. However, worst-case conditions are usually rare, and hence, traditional worst-case design approaches lead to pessimistic design, or in other words to over-design not allowing the full utilization of performance gains obtained due to technology scaling [55]. Furthermore, traditional approaches, i.e., transistor up-sizing or voltage up-scaling increase power dissipation. To circumvent the over-design imposed by the traditional worst-case approaches, statistical timing analysis has been widely investigated [3, 9, 15, 55] at the logic level, where a circuit parameter (e.g., delay or leakage) is modeled as a statistical distribution and the circuit is designed to meet a given yield with respect to a target value of the parameter. Similar to logic circuits, different circuit and architecture level design techniques have been investigated [35, 42] to improve the yield of nano-scaled SRAM as described in previous chapters.

8.1.1.3 Low-Power and Robust Application-Specific System Design

It is evident from the above discussion that low-power and variation-aware design have contradictory design requirements – low power demands down-scaling of voltage, whereas traditional variation-tolerant design requires voltage up-scaling and/or transistor up-scaling. Therefore, there is a need for designers to develop novel design techniques that address the contradictory design requirements simultaneously.

It would be ideal if techniques at the circuit and logic levels could solve the problems introduced by variations. However, there is significant concern in the semiconductor industry that these techniques are not sufficient to contain the increasing impact of variations as circuits scale into the deep nanometer regime [30] [7]. Design techniques available in various levels of hardware stack, such as transistor sizing and architecture modifications (such as parallelism and pipelining) [3, 9, 56] can lead to timing slacks that can be utilized for voltage scaling or tolerance to process variations while ensuring correct operation. However, such techniques come at a cost of area overhead resulting in an increase of power consumption as discussed in previous sub-sections.

Consequently, there has been a lot of interest in recent years [2, 30] to address variations and power consumption early in the design cycle, namely at the algorithm/system level, where it is possible to effectively trade-off variation tolerance, power, and performance with the system level metric of quality-of-results. System level techniques can take advantage of information that is not easily available at the lower levels of abstraction, provide designers feedback about the impact of errors (due to VOS or variations) early on in the design cycle, and facilitate better design decisions. Interestingly, voltage over-scaling and parametric variations are two sides of the same coin. Both of them result in delay failures at the circuit level that translate to logic errors. The logic errors can be seen as computation errors at the architecture/algorithm level and may translate to output quality degradation at the system/application level. Therefore, it could be possible to address the symptoms (delay failures) of VOS and process variations simultaneously at high levels of design abstraction and provide systematic solutions that tackle both.

There exists few research works that jointly address the issues of low-power and parameter variations at the system level [4, 5, 25, 31, 54]. In general, such approaches are either based on the addition of redundant hardware or utilize application-specific characteristics that allow the exclusion of the less-crucial computations that do not influence the output quality significantly. Such classification of techniques for computing under unreliable silicon can be traced back to the work of von Neumann who introduced hardware redundancy to tolerate computational errors [61]. Later Breuer introduced the concept of graceful degradation [11] – rather than dealing with results that are always correct (target of redundancy-based techniques), dealt with techniques to carry out *useful* computations that can lead to *good-enough* output quality. Such approaches target mainly application-specific/DSP applications that differ from general purpose applications since the *quality-of-results* can be traded off for tolerating potential errors [12]. These approaches result in energy efficient designs providing intelligent trade-offs between quality and power while enabling tolerance to parametric variations with minimum area overhead. In general, they take advantage of the fact that for several DSP and multi-media applications, Human Visual/Audio System (HVS) can tolerate imperfect/approximate computations.

8.1.2 Organization

In this chapter, we consider the above computational paradigms in the design of low-power and robust Application-Specific systems in the nanometer era. The techniques differ from approaches that are targeted for general purpose applications such as CRISTA [27] or RAZOR [24] since any performance/throughput penalty induced by these might be detrimental for signal processing applications.

Section 8.2 presents a significance-driven approach (SDA) that identifies *significant/important* computations that are more contributive towards the output quality for a class of applications [32, 34. These computations are subsequently given higher priority through algorithmic and architectural transformations. The application of such approach to various DSP applications is discussed.

In Section 8.3 redundancy-based techniques are discussed [54]. Specifically, algorithmic noise tolerance (ANT) is presented where a reduced precision replica of the original block is added and acts as an error control; it estimates potential errors and tolerates any timing violations by selecting the *most correct* computed output.

Section 8.4 presents techniques based on approximate and stochastic computation. Specifically, error-resilient system architecture (ERSA) [39], scalable computing effort (SCE) [18], and probabilistic design techniques are presented. In addition, alternative computational paradigms [52] are also briefly described. Such approaches provide tolerance to parametric variations while reducing power by treating computations as stochastic processes.

Section 8.5 briefly overviews recent process-aware power management techniques that take into consideration parametric variations, while applying dynamic voltage and frequency scaling [13] to embedded processors for DSP applications.

Section 8.6 discusses design techniques that allow efficient power and quality trade-offs in embedded memory. A design technique that expands the concept of significance-driven design to memory, where 6T and 8T bit-cells are combined into a hybrid memory architecture to provide voltage scaling in memories [16] is presented. The technique uses the concept of unequal error protection at the hardware level – robust bit-cells are employed in MSBs while smaller but less robust bit-cells are employed for LSBs of video computations.

Section 8.7 briefly discusses the need for design techniques that take into consideration system level interaction between the various robust and low-power sub-blocks designed by the above techniques. In addition a mathematical framework [33] that formalizes the exploration of energy efficient design solutions in different levels of hardware stack of various system sub-blocks is presented.

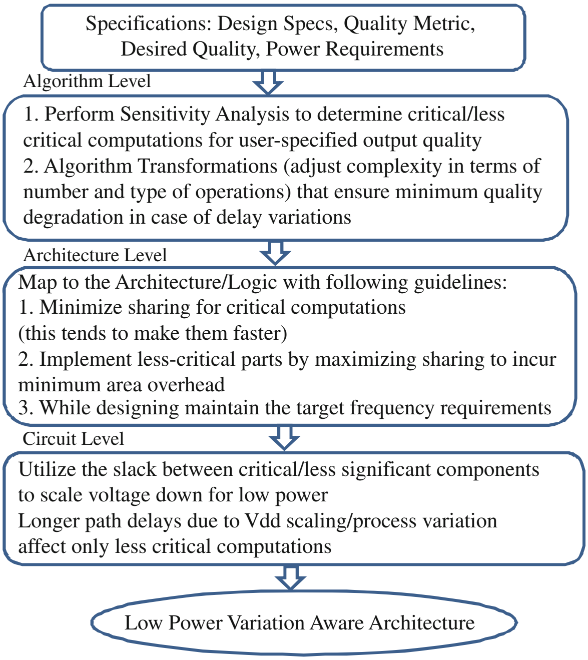
Section 8.8 provides a summary of the presented techniques and concludes the chapter.

8.2 Significance-Driven Approach (SDA) at Design Time

Interestingly, all computations are not equally important in shaping the output response in various DSP applications. For such systems, some computations are critical for determining the output quality, while others play a less important role. This information can be exploited to provide the "right" trade-off between output quality, power consumption, and parametric yield under process variations. Based on such observation a significance-driven design approach (SDA) has been proposed [4, 32, 34].

The design flow of such approach is illustrated in Fig. 8.2. Initially, the target application and user-specifications are determined. Then a sensitivity analysis determines the *significant/less-significant* components. *Significant* are the computations that contribute significantly to the output quality, whereas *less-significant* are the ones that affect the output quality to a lesser extent. Of course, this classification is only possible based on the quality requirements of the user and application specific characteristics. Based on such classification, in the next step, an architecture is developed by giving higher priority to *significant* components. Specifically, the *significant* parts are constrained to be computed within a significantly shorter time than the clock period. This is achieved by co-designing algorithm and architecture. On the other hand the timing constraints of the less significant ones are less strict and can take longer time (within the clock period) making them susceptible to delay failures. Note that the critical or *significant* components share minimal resources since this tends to make them faster. To circumvent the area overhead imposed by the reduced sharing in the *significant* parts, sharing among the less-critical sections of the design is maximized, while maintaining the target-frequency requirements. This maximal sharing for less-critical sections and minimal sharing

Fig. 8.2 SDA design methodology


Specifications: Design Specs, Quality Metric, Desired Quality, Power Requirements
Algorithm Level
1. Perform Sensitivity Analysis to determine critical/less critical computations for user-specified output quality
2. Algorithm Transformations (adjust complexity in terms of number and type of operations) that ensure minimum quality degradation in case of delay variations
Architecture Level
Map to the Architecture/Logic with following guidelines:
1. Minimize sharing for critical computations (this tends to make them faster)
2. Implement less-critical parts by maximizing sharing to incur minimum area overhead
3. While designing maintain the target frequency requirements
Circuit Level
Utilize the slack between critical/less significant components to scale voltage down for low power
Longer path delays due to Vdd scaling/process variation affect only less critical computations
Low Power Variation Aware Architecture

for the critical sections of the architecture leads to a delay slack between *significant* and *less-significant* components. Note that under nominal supply voltage operation, any parametric variations or voltage over-scaling do not have any impact on the *significant* computations. The resultant increase in delay due to voltage over-scaling or parametric variations affects only the *less-significant* computations and has negligible effect on output quality. Other important features of the SDA methodology is the maintenance of the same operating frequency as that of nominal supply voltage at scaled voltages. In other words, using carefully designed algorithms and architectures, the proposed approach provides unequal-error-protection (under voltage over-scaling and parametric variations) to *significant/not-so-significant* computation elements, thereby achieving large improvements in power dissipation with graceful degradation in output quality. The approach has been applied to the design of various voltage scalable and variation-aware architectures, popular in today's multimedia and signal processing systems. Examples include the discrete cosine transform [32, 34], color interpolation [4], FIR filtering [20], and motion estimation [41]. Next the application of SDA to the design of some of these architectures is presented.

8.2.1 Discrete Cosine Transform

One of the most popular DSP architectures, ubiquitous in almost all the multimedia standards (JPEG, MPEG, H.264) is Discrete Cosine Transform (DCT). Conventional 2D-DCT [32, 34] usually is decomposed in two 1-D DCTs which can be expressed in vector-matrix form, as:

$$w = T \bullet x' \tag{8.3}$$

where T is an 8×8 matrix with cosine functions as its elements, and x and w are row and column vectors, respectively [32, 34]. Since 8×8 DCT bases of matrix T in Equation (8.3) have symmetric property, the even/odd 1-D DCT calculation can be re-arranged and expressed as two 4×4 matrix multiplications, in the following manner:

$$\begin{bmatrix} w_0 \\ w_2 \\ w_4 \\ w_6 \end{bmatrix} = \begin{bmatrix} d & d & d & d \\ b & f & -f & -b \\ d & -d & -d & d \\ f & -b & b & -f \end{bmatrix} \begin{bmatrix} x_0 + x_7 \\ x_1 + x_6 \\ x_2 + x_5 \\ x_3 + x_4 \end{bmatrix} \tag{8.4a}$$

$$\begin{bmatrix} w_1 \\ w_3 \\ w_5 \\ w_7 \end{bmatrix} = \begin{bmatrix} a & c & e & g \\ c & -g & -a & -e \\ e & -a & g & c \\ g & -e & c & -a \end{bmatrix} \begin{bmatrix} x_0 - x_7 \\ x_1 - x_6 \\ x_2 - x_5 \\ x_3 - x_4 \end{bmatrix} \tag{8.4b}$$

where $c_k = \cos(k\pi/16)$, $a = c_1$, $b = c_2$, $c = c_3$, $d = c_4$, $e = c_5, f = c_6$, and $g = c_7$. Further decomposition of the above equations result in vector scaling operations which can be easily represented as few shifts and additions:

$$\begin{bmatrix} w_0 \\ w_2 \\ w_4 \\ w_6 \end{bmatrix} = (x_0 + x_7) \begin{bmatrix} d \\ b \\ d \\ f \end{bmatrix} + (x_1 + x_6) \begin{bmatrix} d \\ f \\ -d \\ -b \end{bmatrix} + (x_2 + x_5) \begin{bmatrix} d \\ -f \\ -d \\ b \end{bmatrix} + (x_3 + x_4) \begin{bmatrix} d \\ -b \\ d \\ -f \end{bmatrix} \tag{8.5}$$

In a conventional pipelined DCT architecture, in each clock cycle, each column, $(x_0 - x_7)$ of the input data x is computed and the outputs $(w_0 - w_7)$ (Fig. 8.3b) are stored in a transformation memory (Fig. 8.3c). Then another 8 × 1 column DCT is performed on each column $(y_0 - y_7)$, to obtain all the desired 64 DCT coefficients $(z_0 - z_7)$.

Based on SDA methodology, application-specific characteristic must be identified that will allow graceful degradation of output quality. This property can be easily identified in DCT since signal energy of the DCT output is concentrated on a few low-frequency components (marked as *Significant* in Fig. 8.3d), while most other high-frequency components are associated with small signal energy (marked as *Not-So-Significant* in Fig. 8.3d). Considering that high-frequency DCT coefficients do not carry a lot of energy/information it is expected that overall image quality would not be affected significantly in case that some of them are not computed.

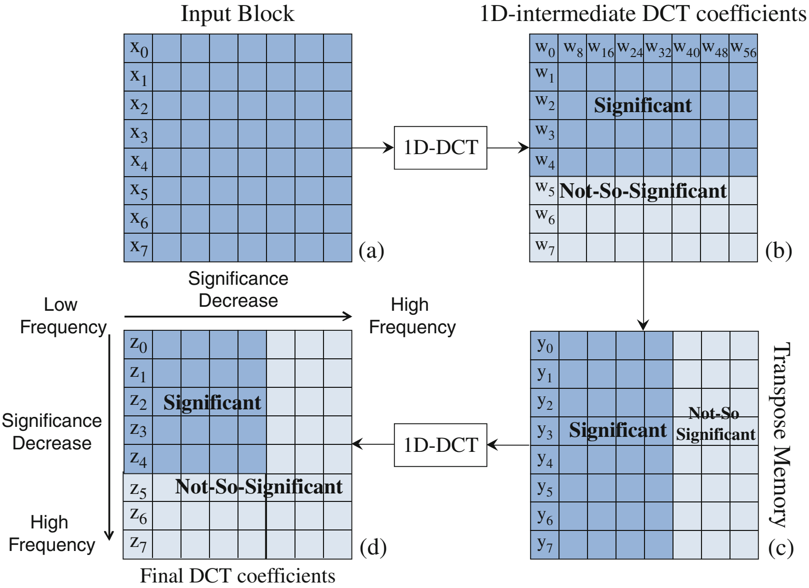


Fig. 8.3 (**a**) Input Pixel Matrix, (**b**) Matrix after first 1-D DCT (**c**) Matrix after being transposed, (**d**) Final 2-D DCT matrix

8.2.1.1 Algorithm

With the above information in mind, the overall SDA-based computational flow for DCT is illustrated in Fig. 8.3. The architecture is designed such that in each clock cycle the *significant* computations (low frequency coefficients, e.g., $w_0 - w_4$ for inputs $x_0 - x_7$, since each w computation takes the entire column values ($x_0 - x_7$)) are protected (by computing them fast such that even under voltage scaling or parametric variations they do not lead to any delay failures) compared to the *less-significant* (high-frequency coefficients) ones, at the cost of more area. The 5×8 sub-matrix is "more significant" than the remaining *less-significant* 3×8 sub-matrix as can be observed in Fig. 8.3b. Since the 5×8 sub-matrix is better protected from delay errors (Fig. 8.3b), the corresponding transpose operation results in "more protected" computation of the 8×5 sub-matrix as shown in Fig. 8.3c. Similarly, the second 1-D DCT operation results in more protected evaluation of the first five (e.g. $z_0 - z_4$) values for each of the input columns ($x_0 - x_7$, etc.).

In order to design such DCT system it is necessary to skew the different path-lengths in the DCT computation. Specifically, the path length of the first five elements ($w_0 - w_4$) of the DCT computation must be significantly shorter than the clock period. This can be achieved by noting the relationship between the DCT coefficients and by slightly altering the values of the original coefficients as shown in Table 8.1. Note that this modification should be performed carefully so that it has minimal effect on the image quality. Specifically it was found through a *sensitivity analysis* that any modification of the coefficient "*d*" degrades the image quality by

Table 8.1 Original and modified 8-bit DCT coefficients

	Original		Modified		
Coefficient	Value	Binary	Value	Binary	Expression
a	0.49	0011 1111	0.50	0100 0000	a
b	0.46	0011 1011	0.47	0011 1100	$e+f$
c	0.42	0011 0101	0.41	0011 0100	$a+e-2\times f$
d	0.35	0010 1101	0.35	0010 1101	d
e	0.28	0010 0100	0.28	0010 0100	e
f	0.19	0001 1000	0.19	0001 1000	f
g	0.10	0000 1100	0.10	0000 1100	$f/2$

a considerable amount, and thus it was defined as the most *significant* component in determining image quality. Therefore, coefficient d is kept unchanged in this process and only the values of coefficients a, e, and f, are modified as shown in Table 8.1. This modification allows the expression of the *less-significant* coefficients b, c, and g in terms of a, e, d, and f.

8.2.1.2 Architecture

Based on these modifications, a scalable architecture for Even and Odd DCT is developed as shown in Fig. 8.4. From Fig. 8.4a–d it is observed that the computations of w_4, w_5, w_6, and w_7 are dependent on w_0, w_1, w_2, and w_3. In terms of path-lengths, w_0 is the shortest since minimal sharing is allowed for fast computation followed by w_1 to w_4, which in turn are shorter than $w_5 - w_7$ (maximal sharing for low power and area reduction). This ensures that the delay in *significant* computational paths $w_0 - w_4$ to be always shorter than clock period even under potential delay failures.

Note that in order to reduce any area overhead imposed due to the minimal sharing employed in the significant parts, multiplier-less techniques can be exploited that represent all multiplications as shift and adds [32, 34]. The resultant architecture of such representation is shown in Fig. 8.4a in case of the path w_0.

8.2.1.3 Voltage Over-Scaling

The delay differences in computational path-lengths are exploited to effectively scale down the voltage and to achieve proper trade-offs between power dissipation and image quality under process variations. The architecture is compared to both a conventional architecture implemented with Wallace Tree Multipliers (WTM) and a multiplier-less architecture based on computation sharing multiplier (CSHM) with six alphabets. Details for the CSHM implementation can be found in [32, 34]. As it can be observed in Table 8.2, the SDA-based DCT architecture allows the scaling of voltage leading to significant power savings.

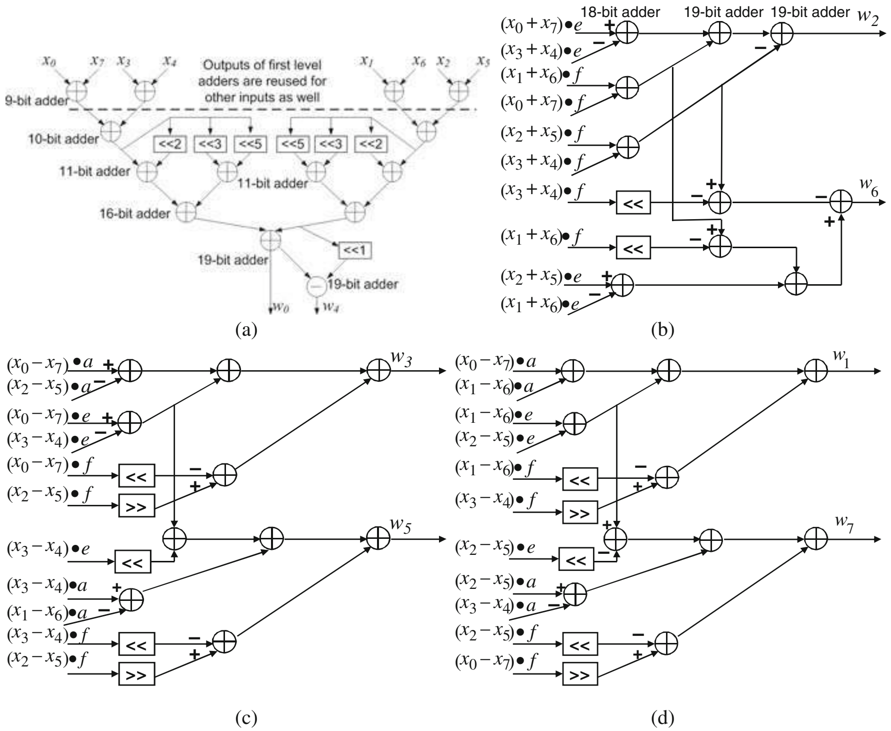
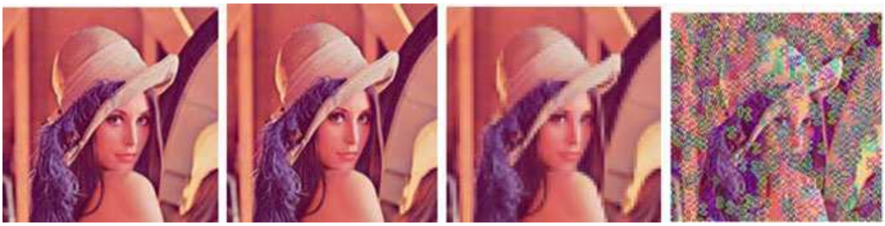


Fig. 8.4 (**a**), (**b**), (**c**), (**d**) show the dependencies among the even and the odd DCT computations based on coefficient relationships

Table 8.2 Comparison of different architectures

$V_{dd} = 1.2$ V	CSHM DCT (6 alphabets)	DCT with WTM	SDA DCT (1.2 V)	SDA DCT (1.02 V)	SDA DCT (0.88 V)
Power (mW)	29.2	32.5	18.59	10.1	5.39
PSNR (dB)	33.22	33.23	33.22	29	23.41
Area (μm^2)	81176	56196	36038		

Figure 8.5 shows the transformed Lena image under scaled V_{dd} in 90-nm process technology. For the SDA-based design, a gradual degradation in the image quality is observed as the voltage is scaled down. Over 71% power savings are achieved when voltage is reduced to 0.88 V, while quality is reduced to an acceptable peak-signal-to-noise ratio (PSNR) of 23.4 dB. Interestingly, the architectural transformations and the increase of computation sharing within least-significant parts allowed by coefficient dependencies resulted in a reduction of area. Such reduced area allowed extra power savings, more than what is expected by VOS. On the other hand, at a V_{dd} of less than 1.17 V, the WTM-based architecture results in significant quality loss. This is because all computational paths in this design are approximately of similar



(a)V_{dd}=1.2V (b)V_{dd}=1.02V (c)V_{dd}=0.88V (d)

Fig. 8.5 Simulation results at (**a**) nominal process corner and V_{dd} (**b**), (**c**) SDA DCT at scaled V_{dd} (**d**) conventional design under scaled V_{dd} or variations

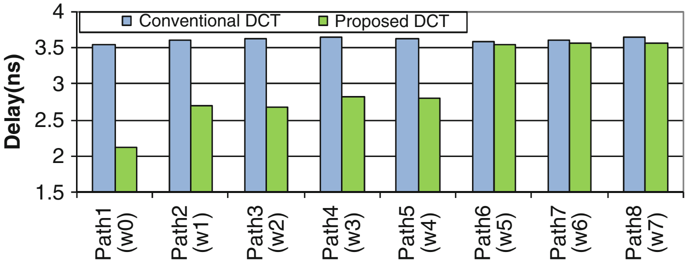


Fig. 8.6 Computational path delays of DCT outputs at nominal V_{dd}

length as shown in Fig. 8.6. Hence, reducing the V_{dd} prevents any complete DCT computation and drastically affects the output image PSNR. Similarly, conventional CSHM architecture is impossible to obtain reasonable image quality below 1.15 V.

8.2.1.4 Parametric Variations

Similar to tolerance of delay failures induced by VOS, SDA-based DCT also provides immunity to parametric variations. Under process variation the delays in the computational paths may vary depending on the process corner that the chip is in. Since, in a conventional DCT design all path-lengths for evaluating the 1-D DCT computations are of almost similar lengths (Fig. 8.6), it is possible that paths ($w_0 - w_4$) contributing to the computation of the high-energy components get affected resulting in significant quality loss (Fig. 8.5d). On the other hand, SDA-based DCT is able to tolerate any errors due to the unequal delay of computational paths depending on their significance. Interestingly, the SDA-based DCT can tolerate errors not only under nominal V_{dd} but also under VOS. Since the *significant* computations are protected in SDA-based design, a fairly high image PSNR can be maintained even when paths w_5–w_7 get affected. This allows achieving ∼99% manufacturing yield for DCT designs with minor quality degradation.

Figure 8.7 presents the trade-offs between complexity/quality levels, power consumption, and process variations (Q1: output obtained from w_0 to w_7, Q2: output

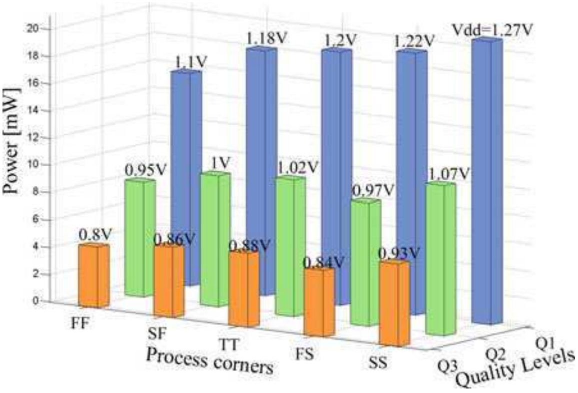


Fig. 8.7 Design space of proposed DCT architecture (Q1 = 34.91 dB (paths $w_0 - w_7$), Q2 = 30.03 dB (paths $w_0 - w_4$), Q3 = 22.34 dB (path w_0))

obtained from w_0 to w_4, and Q3: output obtained from path w_0). In addition, the minimum V_{dd} required for correct operation of the proposed design at various process corners; Typical–Typical (TT), Slow–Slow (SS), Fast–Fast (FF), Fast–Slow (FS), and Slow–Fast (SF) and different computation complexity/quality levels are shown after performing a statistical analysis. Such analysis is crucial for online power management techniques discussed later.

8.2.1.5 SDA at Run Time

The SDA methodology described in Fig. 8.2 is aimed at design time. However, the addition of a low-overhead circuit based on a look-up table can enable on-line detection of process corner and application of suitable V_{dd} depending on quality requirements at run-time. A low-overhead adaptive compensation circuit for voltage/corner detection is shown in Fig. 8.8. The voltage comparators C1 and C2 compare the current supply voltage value with the reference voltage V_{ref1} and V_{ref2} to produce the logic value "1" if the supply voltage is equal to or lower than the reference voltage. The reference voltage V_{ref1} is the value of scaled voltage at which level paths $w_5 - w_7$ fail and V_{ref2} is the scaled voltage at which paths $w_1 - w_7$ fail.

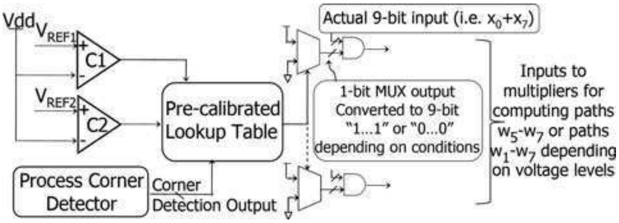

Vdd
VREF1
+
-
C1
VREF2
+
-
C2
Pre-calibrated Lookup Table
Process Corner Detector
Corner Detection Output
Actual 9-bit input (i.e. x0+x7)
1-bit MUX output Converted to 9-bit "1...1" or "0...0" depending on conditions
Inputs to multipliers for computing paths w5-w7 or paths w1-w7 depending on voltage levels

Fig. 8.8 Adaptive compensation circuit

A process detector circuit provides signals indicating the process corner, and a pre-calibrated lookup table contains information about whether a particular path gets affected at a given voltage and process corner. The "multiplier" (implemented with shift and adds) inputs are driven by the outputs of the AND-gates shown in Fig. 8.8. One input of these AND gates is the sum of the primary input pixel values ($x_0 + x_7$, etc.), whose resultant bit-width is 9-bits. The other inputs of these AND gates are generated by multiplexers whose outputs are determined based on the results of the pre-calibrated lookup table (Fig. 8.7). Specifically, the multiplexers force inputs that are associated with less significant computations to zero since they are not required under low-power/parametric variations. This effectively reduces any unnecessary switching activity due to the less-significant computations. Interestingly, since the process detectors and voltage comparators are usually present on-chip/on-wafer, the area/power overhead associated with adaptive quality tuning with voltage scaling and/or parameter discrepancies is low. Moreover, there is no delay overhead due to the single-bit multiplexers during runtime, since the lookup table statically provides the control signals and determines the multiplexer outputs way in advance.

8.2.2 Color Interpolation

The significance-driven approach was also applied to color interpolation [4], which is the most computationally demanding operation in the image color processing stage found in today's digital cameras. To alleviate power requirements as well as to reduce the manufacturing cost, it is customary for image capturing devices to use a single image sensor. In this case, only one of the three primary colors, red (R), green (G), and blue (B), is sampled at each pixel and subsequently stored in color filter arrays (CFA). In order to reconstruct a full color image, the missing color samples at each pixel need to be interpolated by a process called color interpolation. Since the color processing stage output directly impacts the post-processing stage results, it is imperative that a reasonably high PSNR be maintained for interpolation stage output. By allowing "intelligent" trade-offs between image quality and energy consumption, it is possible to develop energy and process-aware interpolation architecture with high output image PSNR.

8.2.2.1 Sensitivity Analysis

Following the methodology described in Section 8.2, the computations required for color interpolation are divided into two parts based on their impact on the output image quality [4]. Based on a sensitivity analysis it was found that *bilinear* terms are absolutely necessary in determining the output quality. Such terms are computed using the average of adjacent pixels of the same color. Therefore, such terms were denoted as *significant* within color interpolation architecture. The second part consists of *correction* terms which are mainly used to improve PSNR and do not affect the output quality significantly. The correction term is defined as a 2-D vector gradient of an image intensity function, with the components given by the derivatives in the horizontal and vertical direction at each image pixel. Interpolation of a missing color value at a pixel is achieved through linear combination of the *bilinear* and *correction* terms. The bilinear and gradient components can be jointly perceived as a 2-D “filter window.” The elements of the 2-D vectors (bilinear and gradient) represent the filter coefficients and the number of elements determines the filter size.

8.2.2.2 Algorithm

The basic idea behind the variation-tolerant architecture is to retain the *bilinear* component under all scenarios (scaled voltages and process variations) and vary the size and complexity of the *gradient* component to achieve low power and variation tolerance. To achieve this, modifications at the algorithmic level are necessary that will ensure minimum quality degradation. By performing a rigorous error analysis, the error imposed after each alteration of the coefficient values/numbers was minimized as shown in [4]. To design an architecture which meets the delay constraints at a reduced voltage, the number of operations and hence the critical computational path of the design was reduced by changing the filter size from 9 to 7 (for G and B at R pixel) and from 11 to 7 (for R and B at G pixel). Similar filter size reduction holds for the other four cases as described in [4]. The new filters retain the bilinear component as in nominal V_{dd} case, and certain terms in the gradient component which result in minimal PSNR degradation are not computed. It should also be noted that essential properties of the interpolation algorithm must be met while modifying it. For instance, the remaining filter coefficients constituting the gradient component must be altered to make their sum equal to zero. This can be attributed to the fact that the gradient calculation requires the computation of derivatives along an intensity direction. Therefore in regions of constant pixel intensities, the gradient should be equal to zero. Under aggressive voltage scaling, only the bilinear component for G and B values at R pixel is obtained. On the other hand, the R and B values at G pixels produce bilinear and a minor gradient component [4].

8.2.2.3 Architecture

Part of the resultant architecture for the interpolation of G at R pixel is illustrated in Fig. 8.9. It can be observed that the algorithmic modifications and reduced sharing allowed the bilinear component to require shorter computational time than the clock period, requiring only two adders. Note that the critical computational paths (CCP) determine the clock period and in this case require four adders and two mux delay in order to be computed. The extra hardware required to incorporate the scaled V_{dd}/process-aware option for all the four outputs are the (eight-bit) three-input multiplexers at the output of the design and two two-input multiplexers at the input side. The multiplexers at the input side allow selection of the different coefficient values that are required at each voltage level. These multiplexers contribute to an overall area overhead of 2% and power overhead of 3% at nominal V_{dd}. Of course, other intelligent circuit techniques can be employed (i.e., V_{dd}/ground gating

Fig. 8.9 Scalable Architecture for the interpolation of (**a**) R channel at G pixel

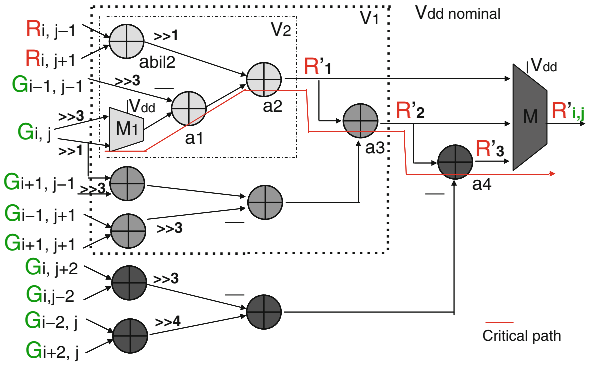


Table 8.3 Power consumption and PSNR under various V_{dd} values

	Nominal V_{dd}		$V_{\mathrm{dd1}} = 0.89$ V		$V_{\mathrm{dd2}} = 0.77$ V	
Method	$P(V)$ mW	PSNR dB	$P(V)$ mW	PSNR dB	$P(V)$ mW	PSNR dB
SDA	13.2	38.15	10.6	35.25	7.9	32.9

of the last stage [50]) to further optimize the area requirements. Depending on the power/quality requirements, the multiplexers select the output that can be computed correctly within the specified delay target.

As seen in Table 8.3, SDA architecture allows voltage over-scaling with graceful image quality degradation and significant power savings. Figure 8.10 presents the trade-off between quality (in terms of PSNR) and minimum voltage required for correct operation of the SDA-based design at various process corners (SS, FF, and TT-typical) in 65-nm IBM process technology. An adaptive compensation circuit described in Section 8.2.1.5 for controlling the multiplexer-enable signals of the

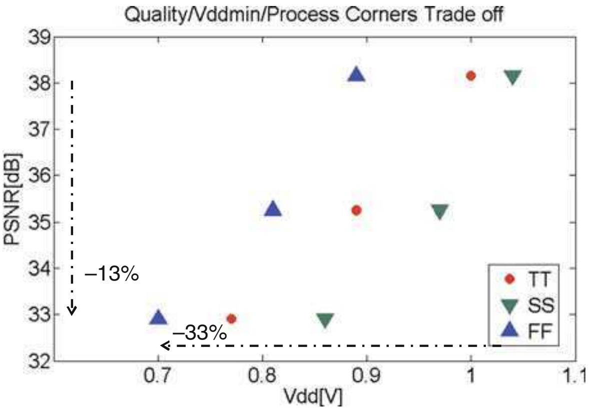


Fig. 8.10 Trade-off quality/Vdd/process corners in 65 nm IBM technology

color interpolation architecture based on V_{dd}/process corner can be used to enable run-time adjustment of the architecture to quality/power requirements.

8.2.3 FIR Filters

The concept of significance-driven computation was also applied to one of the most popular signal processing applications, namely finite-impulse response (FIR) filtering [20]. Specifically, it was found that all filter coefficients are not equally important to obtain a "reasonably accurate" filter response. Based on such observation, a level-constrained common-sub-expression-elimination (LCCSE) algorithm was developed according to which the number of adder levels (ALs) required to compute each of the coefficient outputs can be constrained. By specifying a tighter constraint (in terms of the number of adders in the critical computational path) on the *significant* coefficients, it was ensured that the later computational steps compute only the *less-significant* coefficient outputs. In case of delay variations due to voltage scaling and/or process variations, only the *less-significant* outputs are affected, resulting in graceful degradation of filter quality. The SDA-based algorithm and architecture are discussed in the following paragraphs.

8.2.3.1 Common Sub-expression Elimination (CSE)

Before going into the details of the modified algorithm, computation sharing elimination (CSE) algorithm is described, which is used traditionally for the reduction of the number of arithmetic units (adders and shifters) in low-power FIR filter design. Consider two functions F_1 and F_2, where $F_1 = 13X$ and $F_2 = 29X$. Both F_1 and F_2 can be represented in terms of shift and adds in the following manner: $F_1 = X + X << 2 + X << 3$, $F_2 = X + X << 2 + X << 3 + X << 4$. Both the expressions, F_1 and F_2, have some common terms, $D = X + X << 2 + X << 3$. Therefore, F_1 and F_2 can be rewritten as $F_2 = D + X << 4$. Reusing D in both the expressions reduces the computation overhead and the number of adders required to implement both expressions. The corresponding hardware implementation is shown in Fig. 8.11. Two important conclusions can be drawn from the above example: (i) significant power savings by reducing number of adders and shifters using CSE (only three adders in CSE technique compared to five adders in the unshared case) and, (ii) CSE might increase total number of adders in the critical computational path. We define as critical computational paths (CCP) the paths that determine the clock frequency based on throughput requirements. To elucidate this point further, let us consider each of the expressions, $F1$ and $F2$. Without CSE, $F1$ has two adders in its computational path (Fig. 8.11a). Even after applying CSE, $F1$ is still available after two adder delays (Fig. 8.11b). Therefore, there is no delay penalty for $F1$ in the CSE-based implementation. However, the computational path of $F2$ increases from two to three adders due to CSE, resulting in delay overhead. This shows that there is a trade-off between power consumption and the frequency requirements in case of CSE-based implementation.

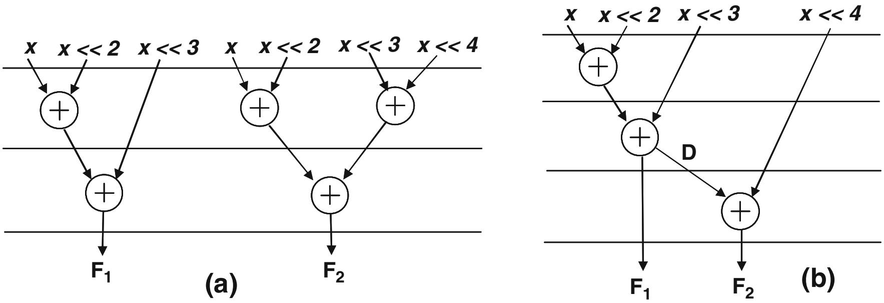


Fig. 8.11 Multiplication (**a**) without CSE (**b**) with CSE

In this point we would like to note that $F1$ and $F2$ can be computed within two adders delay path in the best case. We define such paths as the Minimum Computational Path (MCP) required for the computation of the filter output among all possible realizations. Since the FIR filter coefficients are usually represented in canonic signed digit (CSD) format, the minimum number of adders in their critical path is determined by the number of nonzero terms present in their CSD representation. For example, if a coefficient value is given by "10101001" then MCP consists of two adders. In general, if the coefficient consists of "n" nonzero terms, MCP is given by $\lceil \log_2 n \rceil$. However, as mentioned earlier, due to sharing in CSE a coefficient output might not be computed within the MCP. Therefore, the actual CCP of the whole multiplication block (in traditional FIR filter implementation [20]) can be quite different from the MCP values of the individual coefficients in case of CSE. For instance in Fig. 8.12, though the MCP for both $F1$ and $F2$ is two adders, yet the CCP under CSE is three adders. Based on this observation, three terms are defined:

- Adder Level (AL): AL of a coefficient is defined as the number of adders in critical computational path when the coefficient is computed (AL=3 for "10101001" in Fig. 8.12b);
- Minimum Adder Level (MAL): MAL of a coefficient is defined as the number of adders in minimum computational path among all possible realizations of a

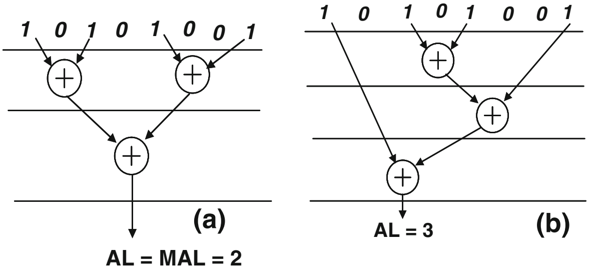
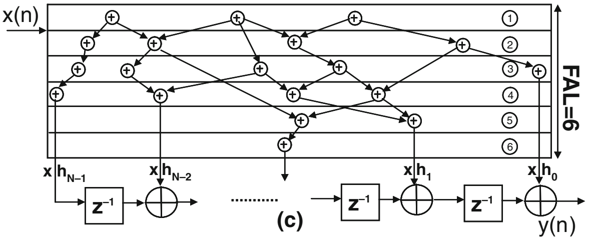


Fig. 8.12 (**a**) Coefficient "1010 1001" realized in AL = MAL = 2 (**b**) Alternative Implementation in AL=3

coefficient and is given by $\lceil \log_2 n \rceil$, where n is the number of nonzero terms in the coefficient (MAL = 2 for “10101001” in Fig. 8.12a).

- Filter adder level (FAL): FAL is the maximum number of adders in the critical computational path of the multiplier block of a transposed form FIR filter. The value of FAL is chosen based on throughput requirements and is user-specified. For instance, in Fig. 8.13, the throughput requirements allow only a maximum of six adders in the critical path of the multiplier block. Therefore, FAL = 6 in this case. It should be noted that the minimum bound on FAL is given by the maximum MAL among all coefficients (FAL $\geq$ max {MAL}).

Fig. 8.13 Filter with FAL = 6



It is also important to note that since the throughput of the filter determines the FAL (and not the MAL), it is not necessary to compute all the coefficients within MAL delay. Moreover, even when all coefficients are computed with their respective MALs, the critical path is determined by the maximum of all the individual MALs. It is interesting to note that as the number of adders in the critical path is relaxed beyond MAL, CSE can be utilized more effectively to reduce the number of adders. For example, if FAL = 2, the common sub-expression is just ($X + X << 2$). On the other hand, FAL = 3 results in more sharing ($X + X << 2 + X << 3$). Therefore, the possibility of sharing the same hardware for those expressions increases, significantly reducing power/area overhead.

8.2.3.2 Significance-Driven Level Constrained CSE (LCCSE)

Having discussed the basics of CSE, in this section the underlying concept utilized in developing a variation-tolerant and low-power design framework for FIR filters is presented. Again the steps shown in Fig. 8.2 are followed. Initially, a sensitivity analysis is performed to identify the most crucial coefficients. The importance of coefficients is determined by the degradation in the pass/stop-band ripples which serve as a metric for estimating the importance of individual coefficients; a larger degradation indicates higher importance.

After identifying the significant/less-significant coefficients, a significance-driven level constrained CSE approach is used to design voltage scalable and variation-tolerant FIR architectures [20]. The concept is explained with an example. Let us consider a transposed form FIR filter where the filter output is computed as:

$$Y[n] = c_0{}^*x[n] + c_1{}^*x[n-1] + c_2{}^*x[n-2] + .. + c_{n-1}{}^*x[1] + c_n{}^*x[0] \tag{8.6}$$

where $c_0, c_1, \ldots, c_n$ denote the coefficients and $x[n], \ldots, x[0]$ denote the inputs. For simplicity, a symmetric filter is assumed where the middle coefficients ($c_{n/2}$, $c_{\underline{n}+1/2}$, etc.) have the largest magnitude. Based on their magnitudes, the coefficients are divided into two sets, *significant* and *less-significant*. After identifying the significant and less significant computations a CSE-based implementation of filter multiplier blocks is exploited and effectively modified. Depending on throughput requirements, if FAL is set to L (also referred to as L levels), then the critical path of the final output from the multiplier block can have a maximum of L adder delays. The *significant/important coefficients* are constrained to occupy a maximum length of L-1 (say), whereas the *less-significant/less-important* coefficients are left unconstrained. This implies that all outputs of the *significant* coefficient set needs to be computed with a path length of L-1 adder (L-1 levels) delays or less. Since the *less-significant* coefficients are unconstrained, the last stage computation (if any) should consist of calculations of the *less-significant* set.

Figures 8.14a, b show a normal CSE-based implementation versus the modified SDA-based CSE design, referred to as level-constrained CSE (LCCSE) [20]. In the normal CSE case, since there is no constraint on the *significant* coefficients, there is a possibility that some of the *significant* coefficients are computed with L adders (L levels). Under a scaled V_{dd} or large process (delay) variation, these outputs might not get computed properly due to delay increase, thereby resulting in large degradation in filter quality. This is prevented in LCCSE-based filters since all the *significant*

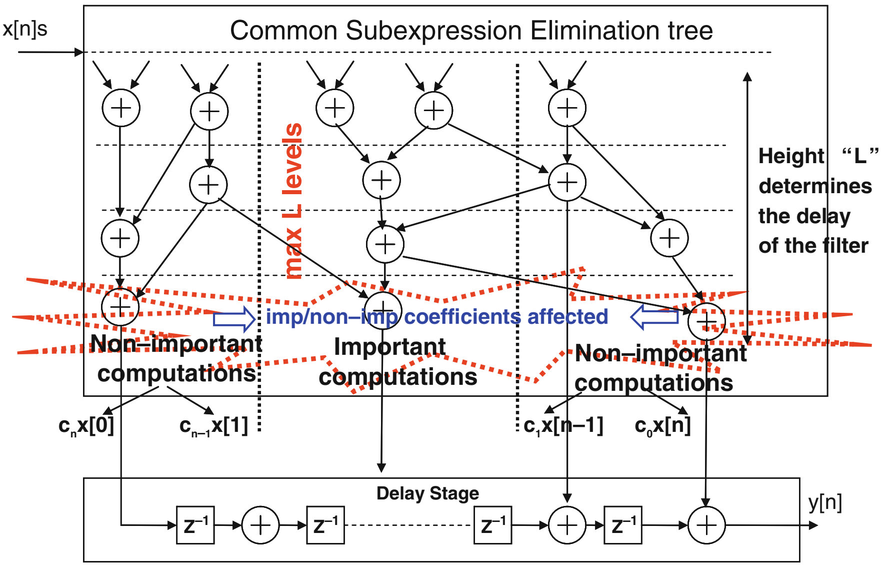

x[n]s
Common Subexpression Elimination tree
max L levels
Height "L" determines the delay of the filter
imp/non–imp coefficients affected
Non–important computations
Important computations
Non–important computations
$c_n x[0]$
$c_{n-1} x[1]$
$c_1 x[n-1]$
$c_0 x[n]$
Delay Stage
z^{-1}
y[n]

Fig. 8.14 (**a**) Synthesis of a FIR filter using conventional CSE. The important computations with longer delays might not be computed under process variation resulting in low filter quality

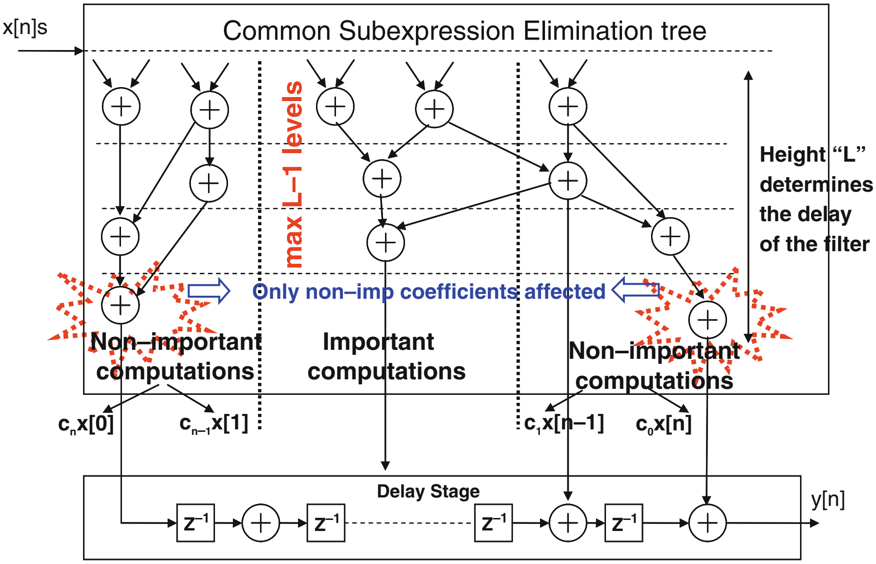

x[n]s
Common Subexpression Elimination tree
max L–1 levels
Height "L" determines the delay of the filter
Only non–imp coefficients affected
Non–important computations
Important computations
Non–important computations
$c_n x[0]$
$c_{n-1} x[1]$
$c_1 x[n-1]$
$c_0 x[n]$
Delay Stage
z^{-1}
z^{-1}
z^{-1}
z^{-1}
y[n]

Fig. 8.14 (**b**) Proposed design methodology where important computations constrained by "intelligent" CSE procedure. Under process variation, high filter quality is maintained

outputs are available one level ahead (for the example considered) of the maximum FAL level (L). In case of delay variations, only the *less-significant* parts are affected. If the coefficient sets are separated into k subsets $\{S_1, S_2, \ldots, S_k\}$ based on their output sensitivities then $\{L_1, L_2, \ldots, L_k\}$ levels can be assigned corresponding to each of the subsets (L_1 for S_1 and so on). Note that L is the FAL specified to provide a certain throughput. This ensures that any coefficient belonging to sensitivity list S_1 should have a maximum path length of L_1, S_2 has a maximum of L_2, and so on. It should be noted that L_i, the maximum AL for the subset S_i, should not be smaller than the maximum MAL among the coefficients belonging to S_i.

LCCSE utilizes a CSE method that considers not only resource sharing among the filter coefficients but also the length of the long paths in the multiplier block. The significance-driven LCCSE algorithm also takes into consideration the level constraints of each coefficient based on its sensitivity. Various level constraints can be given for each coefficient so that tighter timing bounds can be asserted for more *significant* coefficients. Note, in case there is a single level constraint (FAL) for all coefficients, LCCSE yields identical results to conventional CSE.

8.2.3.3 Results

The LCCSE was applied to (i) a 121-tap high-pass FIR filter and (ii) a 32-tap band-pass filter. The coefficients of both the 121-tap and 32-tap filters are divided into three groups with three sensitivity levels (3, 4, 5) based on their relative importance. The adders required for the 121-tap filter implementation are 52 for all 5-level case and 55 for 3/4/5-level case, whereas the 32-tap filter requires 21 and 23 adders for each case. The results of the conventional design as well as the significance-driven designs are summarized in terms of power consumption at nominal and scaled V_{dd}s in Table 8.4. The filter responses for all these cases (conventional and proposed designs under nominal and scaled V_{dd}s) are shown in Fig. 8.15. Under delay variations, the conventional FIR designs cease to operate as seen from the response. Also their pass- band/stop-band ripples increase drastically under such conditions. The proposed FIR design, however, maintains low ripple under parametric variations even at scaled voltages.

Table 8.4 Power (mW) for nominal/scaled V_{dd} for 121/32-tap filter

Power	121-tap high-pass		32-tap band-pass	
V_{dd}	Conventional	Scalable (%)	Conventional	Scalable (mW) (%)
1.0 V	389.9	407.1 (−4.3)	74.5	77.8(−4.4)
0.9 V	fails	341.1(+12.5)	fails	61.1(+18)
0.8 V	fails	277.1(+28.9)	fails	50.2(+32.6)

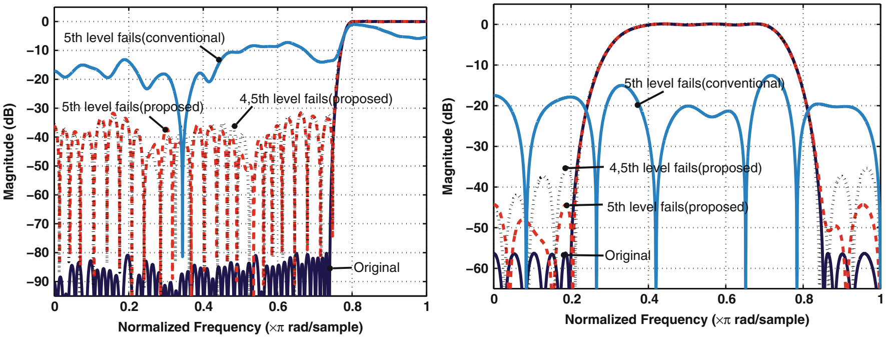


Fig. 8.15 Frequency responses of (**a**) 121-tap high pass FIR filter and (**b**) 32-tap band pass filter

8.3 Redundancy-Based Design Techniques

In fault-tolerant systems, tolerance to errors is commonly achieved by the usage of error detection and correction techniques [36]. Specifically, N-modular redundancy techniques are used. Among them the most common technique is the triple modular redundancy (TMR). In such an approach, the main hardware block is triplicated and then a majority voter compare the outputs between each of the three hardware blocks (main core and redundant blocks) in order to decide the correct output. If any of the three blocks produces an erroneous result compared to the other two then the voter selects the correct output based on voting. In a reliable TMR system, the voter has to be reliable. However, triplication or even duplication of processing units is most of the times prohibitive due to area and power overhead. Therefore, such approaches are only affordable in mission critical systems such as satellite or servers with very high demand on reliability. Influenced by such approaches many researchers [39, 54] have attempted to tolerate errors induced by parametric variations and voltage over-scaling in digital blocks using different forms of redundancy. These approaches mainly attempted to reduce the area overhead associated with error control blocks required in N-modular redundant systems.

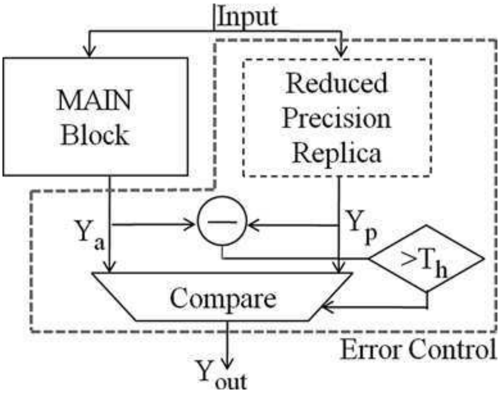


Fig. 8.16 RPR based ANT Design Approach

8.3.1 Algorithmic Noise Tolerance (ANT)

One of the redundancy-based techniques is algorithmic noise tolerance (ANT) [28, 54] which basically permits errors to occur in a main signal processing block and then corrects it via an error control block. An ANT-based system, shown in Fig. 8.16, consists of a main processing unit that correctly computes most of the time, but it is susceptible to parametric variation and VOS induced errors. For instance, the main complex DSP block may be subject to delay failures due to parametric variations and voltage over-scaling. In such cases, the output may not be computed correctly and errors may occur. The output of the main block can be expressed as:

$$Y_{\mathrm{a}}[n] = Y_{\mathrm{o}}[n] + \eta_{\mathrm{n}} \tag{8.7}$$

where $Y_{\mathrm{a}}[n]$ is the main output, $Y_{\mathrm{o}}[n]$ is the expected/error-free output, and η_{n} represents the potentials errors induced by timing violations. The main idea of ANT is the correction of these errors by an estimator that produces a statistical replica $Y_{\mathrm{p}}[n]$ of the error-free main block output $Y_{\mathrm{o}}[n]$. The main challenge in ANT-based systems is to discover a low-complexity estimator with a much smaller critical path delay. This eventually ensures that the estimator output is error-free even though the main block may exhibit timing errors, affected by parametric variations or VOS. Several estimation techniques have been proposed and utilized in the design of ANT-based systems. These include prediction linear prediction (PEC) [28], adaptive error cancellation (AEC) [62], and reduced precision redundancy (RPR) [54].

In each technique, correlation in the signals or cross-correlation between signals and the error is exploited to generate an estimate of the correct output. In general, error detection exploits the fact that in a least-significant bit (LSB) first computation, timing errors in the main block output to occur in the most significant bits (MSBs). Thus, a large deviation between the main block output and the estimated output will be observed during an error event. A simple decision block can be used to detect and correct errors in the main block output as follows:

$$\hat{\mathrm{Y}}_{\mathrm{out}}[\mathrm{n}] = \begin{cases} Y_{\mathrm{a}}[n] \text{ when } \left|Y_{\mathrm{a}}[n] - Y_{\mathrm{p}}[n]\right| < \mathrm{T_h} \\ Y_{\mathrm{p}}[n] \text{ else} \end{cases} \tag{8.8}$$

where T_{h} is a predefined threshold, and $Y_{\mathrm{out}}[n]$ is the corrected final output. The decision block is nothing other a comparator that compares the difference between the outputs obtained from the main block $Y_{\mathrm{p}}[n]$ and from the estimation block $Y_{\mathrm{a}}[n]$. When the difference between the two outputs exceeds a pre-specified threshold, the error-control block declares an error. In the event of an error, the error-control block selects the predictor output $Y_{\mathrm{p}}[n]$. Note that in order to built ANT architectures an optimization process is performed during design time in order to reduce any power and area overhead imposed by the error control blocks.

8.3.1.1 Applications

The various error-control-based ANT techniques (AEC, PEC, and PRP) have been applied to popular DSP architectures. Depending on the error control method applied, the architectures exhibit different power savings (through VOS) and quality degradation.

Specifically, the prediction-based-error-control (PEC) ANT [54] when applied to FIR filters resulted in 67% energy savings over a traditional scaled filter. It has been shown that the ANT-based system improves the SNR by 9 dB when each filter output bit is being flipped at an average rate of one every 1000 samples independent of each other. On the other hand, the adaptive error-cancellation (AEC)-based FIR filters can achieve up to 71% energy savings over critically scaled systems.

However, the method that gain popularity due to its simplicity and better immunity to errors is the reduced precision replica (RPR) based ANT [28, 62], illustrated in Fig. 8.16. While the PEC and AEC are effective for narrowband and broadband systems, respectively, the RPR technique can be applied to both. It was shown that when PRP is combined with VOS, achieves up to 60 and 44% energy savings with no loss in signal-to-noise ratio (SNR) for receive filter in a QPSK system and the butterfly of fast Fourier transform (FFT) in a WLAN OFDM system, respectively. Furthermore, it was shown that the RPR technique is able to maintain the output SNR for error rates of up to 0.09/sample and 0.06/sample in FIR filter and a FFT block, respectively.

Recently ANT was also applied to the design of low-power and error-tolerant Viterbi Decoder [1] used in wireless communication systems as well as to the design of the motion estimation [60] part of video encoders/decoders.

In case of the Viterbi Decoder (VD) the PRP-based ANT scheme was used in order to design the add-compare-select units (ACSU) that are the main computational kernels within a VD. Note that ANT increases latency due to the error control block, which can be a problem for recursive architectures such as the ACSU. Therefore, a lock-interleaved pipelining to increase memory elements in the design, and thereby absorb the increase in latency was also utilized in the design of ANT-based VD [1]. It was shown that such scheme can achieve up to 40 and 25% power savings under VOS and process variations with loss in coding gain of 1.1 and 1.2 dB, respectively, in a 130-nm CMOS process.

On the other hand, in case of motion estimation architecture an input sub-sampled replica ANT (ISR-ANT) of the main sum-of-absolute-difference (MSAD) block was utilized for detecting and correcting the potential errors [60]. Simulations show that such technique can achieve up to 75% power savings over an optimal error-free system in a 45-nm predictive CMOS technology. In the presence of process variations the PSNR in case of ISR-ANT architecture increases by up to 1.8 dB over that of the conventional architecture in 130-nm IBM process technology.

8.3.2 ANT vs. SDA

It is evident from the discussion in Sections 8.2 and 8.3 that both significance-driven (SDA) and ANT approaches result in low-power and process-tolerant architectures. Both techniques require the study and identification of application specific characteristics which can be exploited in order to tolerate any VOS or process variations induced errors. In case of ANT, the challenge is the design of reduced overhead estimators, whereas in SDA the identification of significant/less significant computations based on specific application characteristics is essential.

The basic difference is the area and power overhead imposed in case of ANT. This is evident from the above discussion since ANT is based on redundancy and requires the addition of extra hardware to estimate and tackle any timing errors. Furthermore, the extra comparator block actually increases latency. Such increased latency can be a problem for some recursive architectures such as in the case of the ASCU unit in VD [1] as we discussed.

Considering the power overhead imposed by the extra hardware in case of an ANT-based motion estimation (ME) architecture it was shown that a SDA-based ME architecture [41] yields 15% higher power savings than ANT-based ME architecture at a voltage value of 0.8 V. However, both architectures at the nominal voltage of 1 V, due to the required modifications result in 8% and 30% power overhead compared to a conventional ME architecture. The key difference of SDA from ANT approach is that instead of using replicated hardware running at a slower frequency for addressing delay failures, SDA-based ME relies on prediction of critical path activation to tackle delay failures and perform VOS. Hence, SDA-based ME incurs smaller area overhead and due to lower effective switched capacitance it consumes lesser power than ANT [60].

The same can also be concluded for ANT- and SDA-based DCT architectures [32, 34]. The hardware overhead required in case of ANT design reduces the power savings for ANT-based DCT at a scaled V_{dd} since both the main and replica blocks as well as the comparator circuit operate simultaneously, thereby consuming power.

We would like also to mention that SDA-based designs will require extra hardware in order to ensure low power and process tolerance at the run time due to the addition of a look-up table based compensation circuit, as discussed in Section 8.2.1.5. However, even under this case it is evident that SDA-based designs impose less area overhead compared to ANT-based designs.

It is worth mentioning that ANT-based techniques have also demonstrated their efficacy in providing robustness against soft errors due to particle hits [28]. In this case ANT systems provide robustness and greater energy efficiency compared to traditional TMR-based architectures used for tolerating soft errors.

8.4 Probabilistic and Approximate Computation

ANT and N-modular redundancy based techniques provide tolerance to delay failures by algorithmic and architectural level modifications that attempt not only to estimate but also to "correct" in a way any potential induced errors. However, there are several works that attempt to take advantage of the inherent resilience of various algorithms and instead provide *good-enough* systems without trying to correct any potential errors but rather allowing them to happen. Such techniques take advantage of statistical behavior of the nano-scale devices and circuits and target mainly emerging killer probabilistic applications such as Recognition, Mining, and Synthesis (RMS) applications. Such algorithms typically use iterative and successive refinement techniques, which imparts them with a self-healing nature since subsequent iterations may correct errors introduced in previous iterations. Frequently, these algorithms do not have a single "gold" result; instead, they may produce any one of multiple solutions that are equally acceptable. Of course the usage of such algorithms is such that the user is conditioned to accept less-than-perfect results. SDA is an example of such an approach but in a way it tries also to "correct" any potential error by protecting the significant computations. An overview of such techniques is presented in the next subsections.

8.4.1 Probabilistic Computation

The concept of Probabilistic CMOS or PCMOS, wherein each transistor and logic gate displays a probabilistic rather than deterministic behavior, was proposed as an energy efficient alternative to traditional deterministic computational models [44]. This has led various probabilistic and approximate computation research works, summarized in [46]. Main feature of such works is that they do not attempt to correct errors incurred by circuit elements (as SDA and ANT), instead they utilize them in the context of applications which can tolerate such behaviors. Specifically, a computational device, referred to as probabilistic switch, was introduced that computes one of the four possible 1-bit input and 1-bit output functions with an associated probability of correctness $1/2 \leq p \leq 1$. Logical operations were then modeled as networks of probabilistic switches and later was demonstrated how probabilistic devices may be used in the context of digital signal processing applications [26, 45].

As an extension of such work, a computing platform, referred as stochastic processor, has been proposed in [43]. Such platform targets error-tolerant applications that are able to scale gracefully according to performance demands and power constraints, while producing outputs that are, in the worst case, stochastically correct. Scalability in this case is achieved by exposing to the application layer multiple functional units that differ in their architecture. The various available functional units (i.e., adders) achieve different degrees of voltage/frequency scaling at different error rates. For instance, ripple carry adders employed in a motion estimation architecture of a video encoder can be replaced by Kogge-Stone adders. Based on performance, power, and quality requirements the application may choose between the two adders. In case that 1.2% quality loss is allowed by the application, then the platform can lead up to 60% power savings by dynamically switching between functional-units and dynamically scaling the supply voltage.

8.4.2 Error-Resilient System Architecture (ERSA)

Another approach that takes advantage of the inherent error tolerance of some applications is error-resilient system architecture (ERSA) [39]. ERSA is a low-cost robust system architecture for emerging killer probabilistic applications such as RMS applications. The main problem of such applications is that their quality is significantly degraded due to high-order bit errors and crash when they are executed on error-prone hardware, even if they are resilient to errors in low-order bits of data. ERSA platform is shown in Fig. 8.17. It is a multi-core platform that achieves error resilience to high-order bit errors and control errors in addition to low-order bit errors using a combination of three basic ideas: asymmetric reliability, software optimizations, and error-resilient algorithms. It consists of a limited number of Super Reliable Cores (SRCs) together with a large number of less reliable cores, referred to as Relaxed Reliability Cores (RRCs). The key idea is that the computations of an application are divided into control-intensive (must be error-free) and data-intensive (errors can be tolerated). By applying the idea of asymmetric reliability, the control-related code is assigned to SRCs and the computation intensive code to RRCs. By doing so, ERSA achieves the minimization of processor

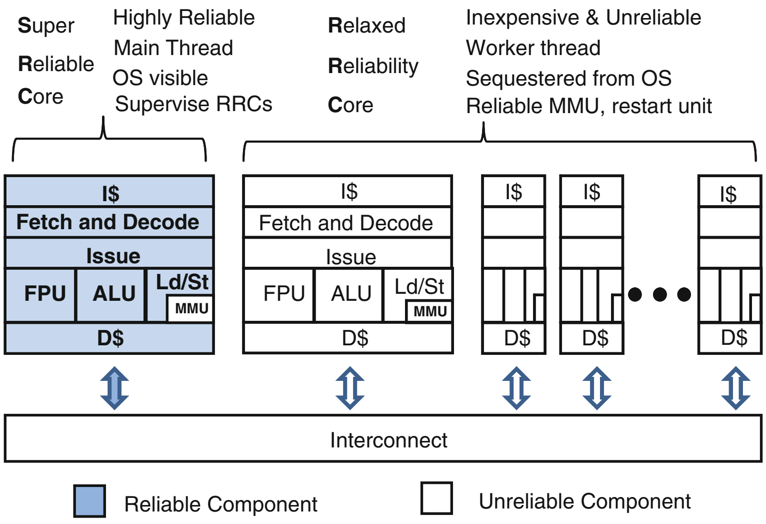


Fig. 8.17 Error Resilient System Architecture

cores that require high reliability, avoiding conservative overall system design. This approach was applied to various probabilistic applications such as K-Means clustering, low density parity (LDPC) decoding, and Bayesian networks using a multi-core ERSA hardware prototype. It was demonstrated that even at very high error rates of 20,000 errors/second/core or 2×10^{-4} error/cycle/core, ERSA maintains 90% or better accuracy of output results, together with minimal impact on execution time. ERSA platforms may also be adapted for general-purpose applications that are less resilient to errors; however these will require higher costs as discussed in [39].

8.4.3 Scalable Computing Effort (SCE)

Another approach that takes advantage of algorithmic resilience of various applications is scalable effort [18]. The basic idea of scalable effort (SCE) design approach is to identify mechanisms at each level of design abstraction (circuit, architecture, and algorithm) that can be used to vary the computational effort expended toward generation of the correct/exact result, and expose them as control knobs in the implementation. These scaling knobs can be utilized to achieve improved energy efficiency, while maintaining an acceptable level of quality of the overall result.

The approach was demonstrated through the design of an energy efficient scalable-effort hardware implementation of Support Vector Machines (SVM) for a popular Machine Learning algorithm. The architecture of SVM platform is shown in Fig. 8.18. A systolic array architecture was used to implement the computation of dot products between support vectors and test vectors, which dominates the workload of SVM classification. The architecture consists of two arrays of FIFOs

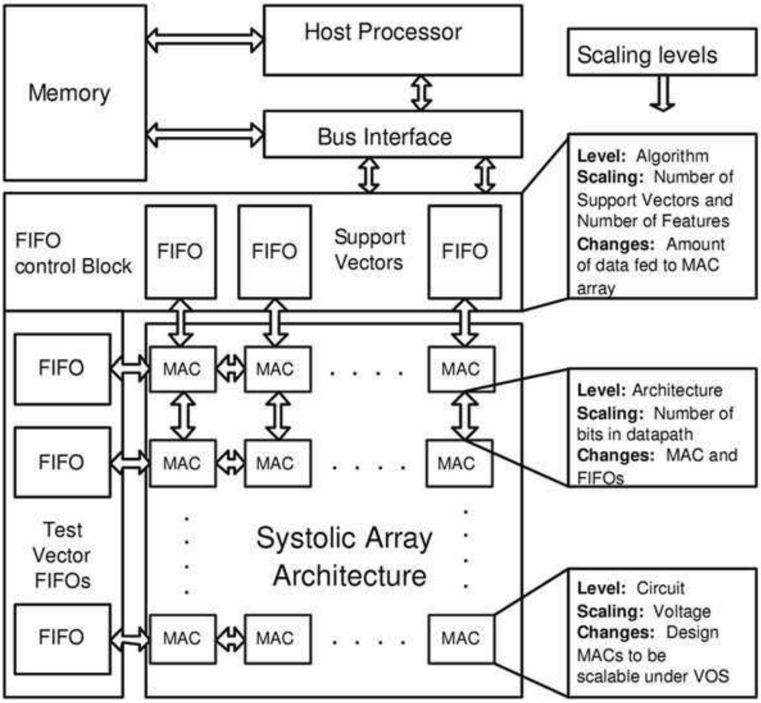


Fig. 8.18 Scalable Effort SVM Architecture

from which data is streamed to a two-dimensional array of multiply and accumulate (MAC) units. Each MAC unit computes the dot product between a unique (support vector and test vector) pair by processing one dimension per cycle in the nominal case and accumulating the result. At the circuit-level, voltage over-scaling was utilized as a mechanism to control the effort expended in order to correctly compute the outputs of computational blocks (MAC) within the clock period, thereby trading accuracy of the result for the energy consumed to compute it. At the architecture level, dynamic precision control of the input vectors was utilized as a mechanism to vary the computational effort expended. Finally, at the algorithm level, a significance-driven algorithmic truncation (by considering the support vectors in the order of their significance) was applied in order to achieve an energy vs. accuracy trade-off.

Results show that scalable effort based platform achieves significant reductions in energy compared to conventional implementations. Specifically, $1.2X$–$2X$ energy reduction was achieved with no loss in classification accuracy, and $2.2X$–$4.1X$ energy reduction achieved with a moderate loss in classification accuracy. Such approach has potential to significantly extend the performance and energy-efficiency of hardware implementations of algorithms in various existing and emerging application domains.

8.4.4 Alternative Computational Models

Alternative models of computation that can tolerate randomness and allow correct operation of future systems have been reviewed in [52]. An example of such approaches is Stochastic Sensor Network on a Chip (SSNoC) [60], described below.

8.4.4.1 Stochastic Sensor Network on a Chip (SSNoC)

SSNoC extends ANT scheme by using distributed computational units (or sensors) whose outputs are combined using statistical signal processing techniques [60]. Traditionally, a main computational block generates a desired output $Y[n]$. However, as we explained the computations in the main block may not be computed correctly due to nano-scale non-idealities. An SSNoC shown in Fig. 8.19 attempts to exploit the robustness of sensor networks to enhance on-chip computation. In an SSNoC, the main computation is decomposed into M lower-complexity sensors with complexity ratio R, where R is the complexity ratio of one sensor to that of the main computation. In this case the sensor outputs $Y_i[n]$ are statistically similar [60], that is:

$$\begin{gathered} Y_i\,[n] = Y[n] + \eta_i[n] \\ E\,\{Y_i\,[n]\} = Y[n],\; E\,\{\eta_i[n]\} = 0 \;\text{ for } 1 \le i \le M \end{gathered} \tag{8.9}$$

Note that statistical similarity implies that the mean of $Y_i[n]$ is a constant across the sensors. The gray shading around some of the black dots in Fig. 8.19 represents the fact that instantaneous sensor outputs $Y_i[n]$ might not equal the correct output. However, such errors in SSNoC are tolerable. Note that $\eta_i[n]$ represents both the estimation error that arises from the usage of low-complexity sensors and the hardware error due to the non-idealities in process and voltage scaling. Irrespective of the error source, a fusion block combines the sensor outputs to produce an output $Y_f[n]$, which is statistically close to the correct output $Y[n]$. The fusion block basically

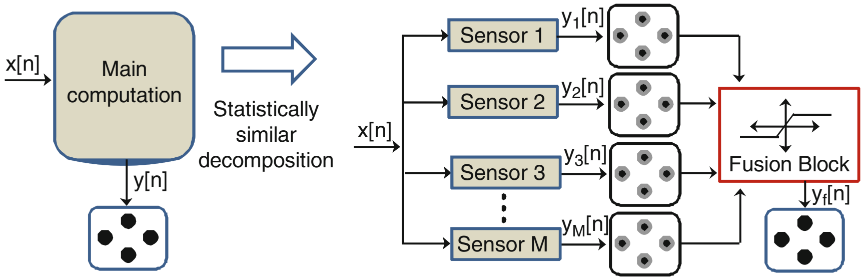


Fig. 8.19 Stochastic Sensor Network on a Chip

implements an algorithm known as One-step Huber algorithm which can compute the parameters of an estimator as described in [60].

The complexity of an SSNoC depends on the complexity ratio R, the decomposition factor M (number of sensors), and the implementation of the fusion block. Interestingly, in case that $R = 1$ and the fusion block is a majority voter, then SSNoC is equivalent to N-modular redundancy (NMR). Note that in case that $R = 1/M$, the only hardware overhead in SSNoC is the fusion block. Therefore, efficient implementation of fusion block is crucial in such approach in order to reduce the overhead. In general, as R decreases, the estimation error (due to low complexity sensors) increases and hardware error (due to parametric variations) usually decreases.

SSNoC concept was applied to some applications such as a pseudonoise (PN) code-acquisition matched filter that is typically employed in code division multiple-access (CDMA) wireless systems [60]. Performance of the SSNOC-based PN-code acquisition architecture at the slow process corner indicates that the average probability of detection improves by up to three orders-of-magnitude over that of the conventional architecture, while the variation in detection probability (σ/μ) is reduced by up to two orders-of-magnitude over that of the conventional architecture. SSNoC in this case resulted in 39% power reduction. There are a lot of issues that need to be investigated for such approach still. Its applicability to generic computational applications and other media blocks is anticipated.

8.5 Variation-Aware System Level Power Management

At the system level, power management is one of the most widely used power reduction technique. However, the conventional power management schemes, designed using nominal power characteristics, can result in substantial power wastage and fail under parametric variations [5, 25, 31]. Therefore, effective power management techniques that can lead to significant energy savings even in the presence of parametric variations are required. Several researchers have proposed power management techniques [13, 29, 49, 53, 63] for complex multiprocessor devices that attempt to efficiently utilize the available resources and assign tasks based on throughput and power requirement while considering process variations. Such approaches, in general, exploit the variable workload of applications over time in order to adjust the voltage in the various power domains on chip. In addition software optimizations that explore data locality existing in almost all DSP and multimedia applications is exploited in order to achieve low power and robustness in embedded processors [48] .

To provide an insight in this area of research we overview two representative variation-tolerant power management techniques proposed in [13] and [31], namely, *design-specific* and *chip-specific* approaches. In each of these approaches, the goal is to consider the impact of variations while deriving power management policy parameters, in order to optimize metrics that are relevant under variations.

In the *design-specific* approach, a set of values for parameters that control the power management scheme is obtained at design time and fixed across all fabricated instances of the chip. The difference from conventional power management schemes is that these parameter values are computed with an objective to optimize the resulting power or energy distribution across all chip instances. In *chip-specific* approach, parameter values are determined for each chip instance based on its individual characteristics. Specifically, each chip senses its power characteristics after fabrication and configure the power management parameters accordingly. To better capture the effectiveness of a power management technique, metrics that consider higher moments of an energy distribution (rather than deterministic values), such as $\mu + \sigma$ and the *N*th percentile of the energy distribution, were used.

Note that both shutdown-based and slowdown-based power management techniques were considered. In shutdown based dynamic power management (DPM), a power manageable component (PMC) is put into low power states during periods of inactivity. These transitions are governed by power management policies, such as timeout-based, history-based, and stochastic policies. On the other hand, in slowdown-based power management or dynamic voltage scaling (DVS), slack in the workload is exploited to allow execution at the lowest possible frequency level such that the performance targets are met. This allows the supply voltage to be scaled down in order to reduce energy consumption.

A variation-aware DVS scheme called variable discrete DVS was also proposed in [5] according to which voltage levels are determined at design time while taking variations into account. At runtime, the voltage level for a given frequency level is selected based on the frequency–voltage characteristics of each chip instance. The proposed schemes were experimentally evaluated in the context of an ARM946 processor core. For the oracle-based framework, variation-aware power management can result in improvements of up to 59% for $\mu + \sigma$; for 95th percentile of the energy distribution, over conventional power management schemes that do not consider variations. For the timeout-based power management, reductions of up to 43% in $\mu + \sigma$ and up to 55% in the 99th percentile of the energy distribution were obtained.

8.6 Memory Design for Power-Quality-Yield Trade-Offs

Apart from logic computations ubiquitous part of today's embedded systems is memory. Therefore design techniques that address the requirements of low power and robustness to parametric variations in memories are required. Conventionally, circuit techniques such as sizing of cell transistors and adaptive body biasing (ABB) [42] and architectural techniques such as addition of redundant columns and rows [50] and use of parity bits for error detection and protection (ECC) [26] were employed for the design of low power and robust embedded memories. In an attempt to limit the area overhead of such techniques, memory design techniques have been proposed that exploit the inherent algorithmic resiliency in DSP applications leading to significant power savings with graceful quality degradation [19, 37, 38].

One of such approaches [16] is based on SDA methodology, according to which circuit and architectural techniques are combined in order to ensure that significant bits are not corrupted under parametric variations and voltage over-scaling. The basic idea lies behind the observation that 8T SRAM cells are more robust than their 6T counterparts at scaled supply voltages. This feature was exploited to design a hybrid memory for video applications based on a mixture of 8T and 6T SRAM bit-cell configuration, employing a low-overhead preferential storage, where the *significant* computations are stored in the more robust 8T-cell memory, while the *less-significant* computations are stored in the 6T-cell memory. The fundamental premise of this approach lies in the fact that human visual system (HVS) is sensitive mainly to higher order bits of luminance pixels in video data. This property of video data was exploited to implement the higher order luma bits as 8T bit-cells, while the lower order five-bits were stored in 6T bit-cells. A sensitivity analysis of video quality on the number and type of bits revealed that three MSB bits are sufficient and must be protected in order to obtain good output quality.

Figure 8.20 shows the schematic of 6T and 8T bit-cells for the hybrid SRAM. It should be noted that the 8T bit-cell has two word lines, which are for read (RWL) and write (WWL), respectively. For efficient integrated layout, the word line of the 6T bit-cell was split to WWL and RWL. Since the hybrid SRAM uses single-ended sensing method, successful read operation can be achieved with one access transistor in the 6T bit-cell. It should be noted that the "BL" node of a 6T bit-cell is used for both read and write unlike that of an 8T bit-cell. The array (32 Kb) of the hybrid SRAM consists of four decoding blocks, and each block has eight sub-arrays. Each sub-array shares local read bitlines (rbl#[7:0]), which are NAND-gated with those of other neighboring sub-arrays. The outputs of the NAND-gates are fed to the gate inputs of the following NMOS transistors connected to global read bitlines (grbl#[7:0]). According to the data values of read bit-cells, the pre-charged local read bitlines and global read bitlines are evaluated sequentially. To ensure that SRAM can be operated at high frequency (~MHz) the number of bit-cells on a sub-array column were determined to be 32.

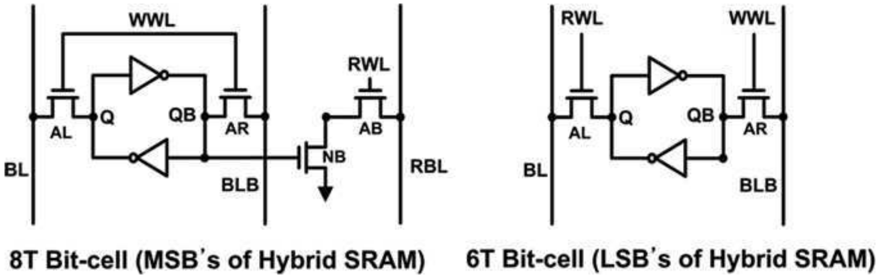

WWL
AL
Q
QB
AR
BL
BLB
RWL
AB
NB
RBL
8T Bit-cell (MSB's of Hybrid SRAM)
RWL
WWL
AL
Q
QB
AR
BL
BLB
6T Bit-cell (LSB's of Hybrid SRAM)

Fig. 8.20 6T and 8T bit-cell schematic for the hybrid SRAM array

Such approach facilitates aggressive voltage scaling in memory as the significant luma bits, stored in 8T bit-cells, remain unaffected by VOS. The less-important lower order luma bits stored in 6T bit-cells, despite being affected by the voltage scaling, contribute insignificantly to the overall degradation in output video quality. Results showed that SDA-based hybrid SRAM array dissipates almost comparable power to the 6T-only array at 800 mV V_{dd}. However, the voltage over-scaling allowed by the hybrid memory resulted in significant power savings for both read and write (44∼46% at 10 MHz frequency) operations. Specifically, hybrid scheme allowed scaling of operating voltage of memory to as low as 600 mV (∼200 mV more scaling than the allowed supply voltage of conventional 6T SRAM in 65 nm) without degrading the output video quality significantly under different process corners as shown in Fig. 8.21. As it can be seen in Fig. 8.21a, 6T bit-cells suffer from drastic read failure probability and hence, frame quality is significantly degraded, at fast-slow (FS) corner, or a V_{dd} of 600 mV. On the other hand, SDA-based SRAM allowed operation under aggressive voltage over-scaling (at 600 mV) with insignificant image quality loss. Specifically, the video quality in terms of PSNR for the proposed hybrid SRAM array was 22.80 dB (Fig. 8.21b) at the FS corner (worst-case corner for 6T cells) and 23.04 dB (Fig. 8.21c) at the SF corner (worst-case corner for 8T cells) which is comparable to that of the 6T-only array at 800 mV (23.38 dB). Furthermore, the hybrid nature alleviates the area overhead issue associated with a full 8T SRAM array resulting in only 11% area overhead when the three significant luma bits and all motion vectors are implemented as 8T bit-cells.

We would like also to mention that such type of memory design that employs voltage over-scaling is aimed mainly for applications that require low operating frequency. For instance, in this application, CIF/QCIF video formats were chosen that require low operating frequency (below 10 MHz) which can be easily satisfied in the 65-nm technology (even with 600 mV of V_{dd}). Specifically, results showed that at 1 V V_{dd}, the hybrid memory can be operated at 600 MHz, whereas at V_{dd} of 600 mV, 50 MHz performance can be achieved satisfying the CIF/QCIF requirements.

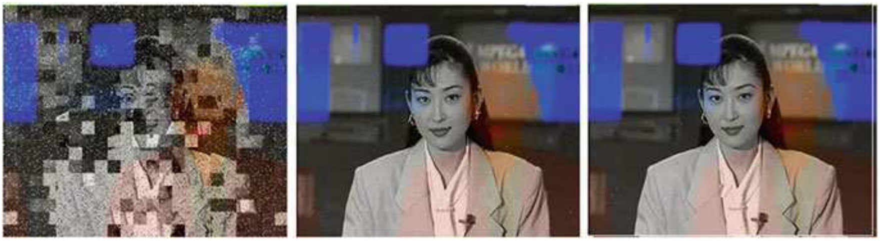


Fig. 8.21 Output video image of 32 Kbit (**a**) SRAM array with 6T-only cells at FS corner and $V_{dd} = 600$ mV, PSNR = 15.27 dB, (**b**) hybrid SRAM array at $V_{dd} = 600$ mV and FS corner, PSNR = 23.61 dB and (**c**) hybrid SRAM array at $V_{dd} = 600$ mV and SF corner, PSNR = 23.90 dB

8.7 Energy Efficient System Design

The majority of the design methodologies presented in this chapter such as ANT and SDA are mainly destined for the design of low-power and variation-tolerant DSP blocks, while trading-off quality-of-results. While these methodologies have proven to be effective in the design of individual voltage scalable blocks, they may be inadequate when these blocks are integrated within a system. Moreover, in case of system level techniques such as SCE and stochastic processors there is a need to identify the most energy efficient solutions at various levels of design hierarchy. In contrast to lower level variables that affect the circuit metrics (energy and delay) of each block (micro-level), algorithmic transformations in one block can potentially affect computations in the other blocks (macro level) due to system level interactions. Interestingly, while some blocks may operate at lower energy and achieve "good" block level quality, subsequent blocks might still be consuming higher power without improving the system quality. In other words, there is a need to determine the "right" energy-quality budget for each block (under a delay constraint in each block) of the whole system and subsequently exploit the tuning variables (in different levels of design abstraction) with the largest capability for energy reduction, considering also that any block level decision will affect the whole system.

8.7.1 System Level DSP Synthesis

To that effect a design methodology for the design of DSP blocks while considering system level interactions has been proposed [32, 34]. Such methodology was applied to the design of SDA-based blocks. The main idea is that voltage scaling in some blocks may lead to incorrect computations, which may affect computations of subsequent blocks that are associated with such potential incorrect computations. In this case there is no need to compute the incorrect computations. Rather the subsequent blocks can be designed such that under VOS only the less-significant computations are affected. Note that in system level, less-significant are the computations that are associated with potential incorrect computations due to VOS. Such methodology allows further reduction of power at minimum quality degradation since all blocks are designed such that only the less-significant computations are affected. The methodology was applied to the design of a multimedia system composed of the main blocks of a digital camera or video encoder. It was shown that a sub-system (Fig. 8.22) consisting of DCT, IDCT, and quantize blocks achieves large power benefits (up to 69%) at reasonable image quality, while tolerating errors (η_A, η_B) induced by varying operating conditions (VOS, process variations, and channel noise).

The methodology can be extended and incorporate blocks designed by other techniques such as ANT and lead-embedded portable systems leading to low power and robust systems for various applications, multimedia, and communications.

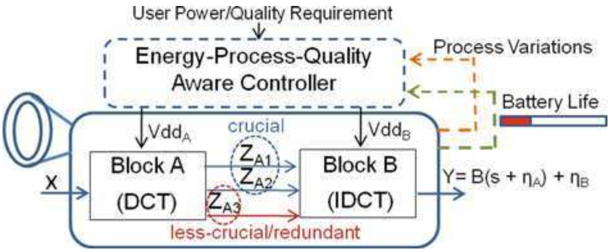


Fig. 8.22 Power and Process Aware System Design

8.7.2 Optimal System Level Energy Efficient Design

Techniques such as scalable effort, stochastic processors, and ERSA target the system level and explore solutions at various levels of design abstraction. However, there is no formal mathematical formulation to allow the identification of the energy efficient solutions that lead to optimal energy-quality trade-offs. To that effect a cross-layer optimization framework for block/system design space exploration, connecting system quality and algorithm implementation with circuit energy and supply voltage scaling was proposed, referred to as Herqules [33]. In Herqules, the problem of minimizing the energy of a system (E_S) that consists of N blocks, meeting the user quality (Q_{DES}) and delay requirements ($D_{\mathrm{TARG,i}}$) of each block was formulated as:

$$\begin{aligned} \min_{\underline{V},\underline{G},\underline{R}} E_S\left(\underline{V},\underline{R}\right) &= \sum_{i=1}^{N} E_i\left(v_i, r_i\right) \\ \text{sbj. to: } Q_S\left(\underline{G}\right) &= Q_N\left(g_i, ..., g_N\right) \le Q_{\mathrm{DES}} \\ D_i\left(v_i, g_i, r_i\right) &= D_{\mathrm{TARG},i} \quad i = 1, 2, ..., N \end{aligned} \tag{8.10}$$

The above formulation captures the interactions of circuit metrics (energy E_i, delay D_i) of each block i with system metric (quality/error Qs) and their dependence on various design techniques. Specifically, v_i, g_i, and r_i represent the various tuning variables available in circuit (such as VOS), architecture (such as type of adders, parallel, or pipeline implementation), and algorithm (such as modification of bit-width and number and values of coefficients) level of each block i, respectively. Herqules taking into consideration the interactions between different sub-blocks of a system; it identifies the design solutions that can ensure the least energy at the "right amount of quality" for each sub-block/system under user quality/delay constraints. This framework introduces new sensitivity-based metrics derived by solving the generic energy-quality-delay optimization problem (Equation 8.10) applying the Karush-Kuhn-Tucker (KKT) theorem [21]. The satisfaction of the derived metrics can ensure block and system level energy efficient operation. Specifically the block (intra) level optimality criteria show that energy efficient operation in block level can only be achieved if the design solutions available at different levels of the hardware stack satisfy the criteria:

$$\frac{\partial E_i}{\partial v_i}\frac{v_i}{E_i}\Big/\frac{\partial D_i}{\partial v_i}\frac{v_i}{D_i} = \frac{\partial E_i}{\partial r_i}\frac{v_i}{E_i}\Big/\frac{\partial D_i}{\partial r_i}\frac{r_i}{D_i}, \quad i = 1,\ldots,N \tag{8.11}$$

The above criteria dictate that the marginal costs (energy and delay) of tuning variables (voltage and architecture) must be balanced to achieve energy-delay efficiency. Such factors are nothing but the absolute gradient of energy savings to some delay decrement, depicted in Fig. 8.23a. For instance, *voltage intensity* is defined to represent the energy-delay trade-off achieved by tuning supply voltage *v*:

$${}^{E}_{D}S_v = \frac{E_v}{D_v} = \frac{dE}{dv}\frac{v}{E}\Big/\frac{dD}{dv}\frac{v}{D} \tag{8.12}$$

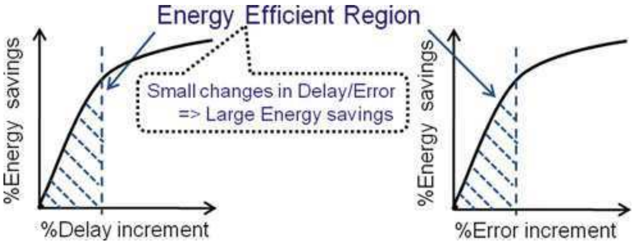


Fig. 8.23 Energy efficient regions with respect to (**a**) % delay increment, (**b**) % error increment

Such factors can effectively represent and evaluate the energy savings achievable by tuning various variables after relaxing any constraint – delay or quality (error). Using such factors we can actually identify the energy efficient regions of operation (Fig. 8.23), where any delay/error increment results in larger return in energy savings. Within an energy efficient region the gradient (i.e., *voltage intensity*) is large, meaning that the design is more sensitive to changes within that region compared to less efficient regions. In other words, The above criteria dictate that *architecture intensity* (energy-delay trade-off due to modified r_i) must equal the *voltage intensity* (energy-delay trade-off due to modified v_i). In case that *voltage intensity* is less than *architectural intensity* then different architecture implementations (say different type of adders) could be selected to achieve better energy/delay ratio. Such criteria can be generalized and effectively evaluate other design solutions. For instance, in the energy minimization problem of a block, where the only constraint was the delay [22, 40, 66] authors introduced the term *hardware intensity* to capture the energy and delay sensitivity with respect to transistor sizing. They show that the design of energy efficient pipelined blocks can be achieved when *voltage* and *hardware intensity* are equal. Therefore the work in [33] generalizes the energy-delay optimization problems including higher level metrics and design techniques denoting that in order to achieve energy efficiency (in block level), designer has to search for the design options that equalize the *architecture*, *voltage*, and *hardware* intensities.

Apart from optimality criteria for energy efficient operation in block level, the framework derives also criteria necessary to be satisfied for energy efficiency in system level:

$$\frac{E_i}{E_{\mathrm{s}}}\frac{1}{Q_i}\frac{D_{g,i}}{D_{v,i}}{}^{E}_{Q}S^{i}_{v,g} = \frac{E_j}{E_s}\frac{1}{Q_j}\frac{D_{g,j}}{D_{v,j}}{}^{E}_{Q}S^{j}_{v,g} \ \forall\text{i,j block} \tag{8.13}$$

Authors in [33] defined the term:

$${}^{E}_{Q}S^{i}_{v,g} = \frac{E_{v,i}}{Q_{g,i}} \tag{8.14}$$

as *algorithm intensity* to represent the trade-offs between energy and quality, achieved by tuning the circuit v and algorithmic g variables within each block. In addition the left hand side of the expression 8.13 is defined as *system intensity* z_i and it links the *algorithm intensity* to architecture and circuit design techniques. z_i consists of the *algorithm intensity*, weighted by the block's contribution to overall system energy and output quality and the ratio of timing gains achieved by tuning g, and v. The above criteria imply that the weighted *algorithm intensity* of a block i should be balanced among all the blocks of a system. This leads ultimately to the point where the energy reduction potential of all variables in each blocks is balanced. In other words, the balance of *system intensity* among all blocks, along with the block level optimality criteria described by equation 8.10 lead to minimum system energy at the right amount of quality, while meeting each block's target delay and system quality. Such criteria can be utilized as design guidelines in early stages of design flow to determine the efficient solutions at different layers of design abstraction.

Herqules when applied to the exploration of energy-quality design space of the main blocks of a digital camera and a wireless receiver achieves 58 and 33% energy savings, respectively, compared to reference non-optimal designs.

Specifically, the design methodology was applied to a camera sub-system consisting of color interpolation (CI) and DCT blocks presented in Section 8.2. The energy-delay-quality trade-offs achieved by considering various types of adders and degree of voltage scaling were explored. Following the framework in [33] the *system intensities* for each block were calculated as shown in Fig. 8.24. Large *system intensity* for a block means that any change in the specific block provides more energy savings for the given error compared to changes in other blocks. We can observe that *system intensity* of DCT is larger than the *system intensity* of CI, for larger range of error increments. This is valid since DCT is a larger block and any VOS will lead to larger energy savings compared to CI.

Figure 8.25 illustrates potential system energy trade-off (achieved by combing the E–Q ratios of each block for different design points. Figure 8.25 also depicts the energy-quality trade-off curve of the system, which represents the envelope of the

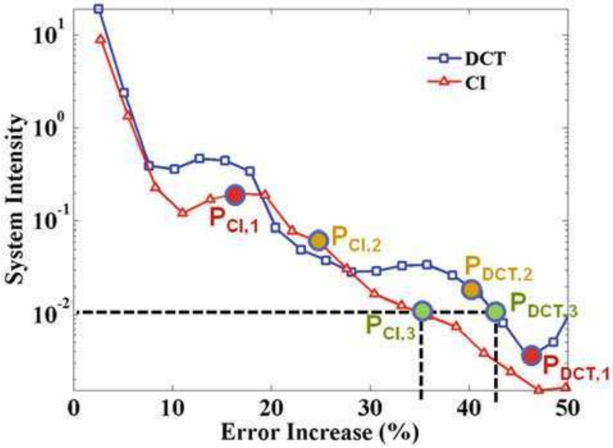
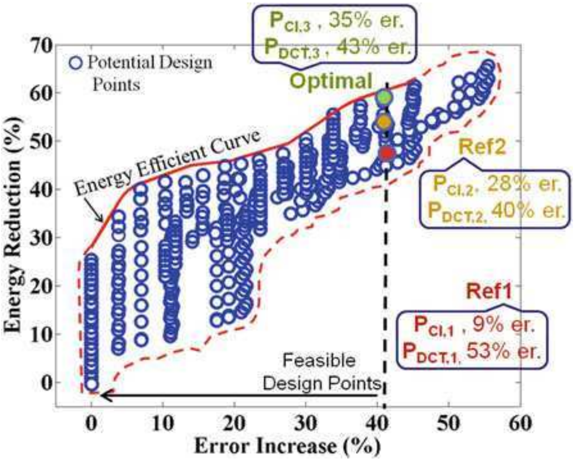


Fig. 8.24 System intensities for CI and DCT

Fig. 8.25 Optimal camera E-Q trade-off block



potential design points. Note that the points on that curve correspond to the design points at which the *system intensities* of the blocks are balanced. Such points either yield the smallest energy of all points with same error, or equivalently they lead to the smallest error among all points with same energy. Let us assume that the user relaxes the system quality (Q_{DES}) accepting a 41% increase in system error. The potential design points that lead to 41% error increment can be easily determined from Fig. 8.25. Intuitively, designers will select any of the design points that provide 41% error increment and achieve energy savings. For instance, someone could choose to increase error by 19% in CI and by 52% in DCT. On the other hand someone else could choose to increase error by 28 and 40% in CI and DCT, respectively. The common characteristic of the above design points is the imbalanced *system* *intensities* (z_{CI} and z_{DCT}) as shown in Figs. 8.24 and 8.25. In both cases the system will be operating under inefficient conditions, consuming more energy than what is required for providing the desired quality. The imbalance between z_{CI} and z_{DCT} denotes that there is significant room for improvement by tuning the knobs of each block. The energy efficient curve obtained after balancing z_{CI} and z_{DCT} can be used to determine the tuning degree of each variable within CI and DCT. It can actually identify the optimal energy-quality budget of each block that leads to the least energy under the given quality constraint. Specifically, the design point that lies on the energy efficient curve (Fig. 8.24) dictates that an error increment of 35 and 43% in CI and DCT, respectively, will lead to 58% system energy savings under the given quality constraint.

Such framework can be valuable in the design of complex systems as described by ERSA, SCE, and approximate computation approaches and can easily incorporate other sources of delay failures since the criteria were deduced without specific error and delay metrics.

8.8 Summary and Conclusion

Technology scaling allowed unprecedented growth of semiconductor industry and the integration of more complex systems in small portable devices. However, scaling faces severe problems due to increased short-channel effects, high leakage power consumption, and parameter variations. While traditional approaches at the circuit and logic level may help alleviate some of the problems, they fail to address both low power and variation tolerance, simultaneously. We believe that system level techniques that provide cross-layer optimization can provide intelligent trade-offs between circuit metrics such as power and performance with system level metrics such as quality-of-results and tolerance to variations (yield). The merit of such approaches is that they can take advantage of information that is not easily available at the lower levels of abstraction, provide designers feedback about the impact of errors (due to VOS or variations) early on in the design cycle, and facilitate better design decisions. By employing low-overhead mechanisms, systems can easily adapt to online changing conditions and cope with variations and voltage scaling induced errors simultaneously. For instance, in case that user requires low-power operation, the voltage level of each block of the system can be adjusted to a value that will lead to low power consumption with minimum impact on output quality. This can be achieved by exploiting the nature of the algorithm and ensuring minimal impact on the quality of service (QoS) of the system under potential errors as discussed in SDA and ANT approaches.

The combination of techniques presented in this chapter, summarized in Table 8.5 can effectively allow the design of low power and robust integrated systems in the nano-scale as well as in the post-silicon era. For instance, SDA- and ANT-based blocks could be combined and extended to ERSA-based platform, assigning *significant* computations to reliable SRCs and *less-significant* ones to the less

Table 8.5 Summary of system level low-power and process-aware techniques

Technique	Mechanism	Advantages/disadvantages
SDA	–Identify significant and less-significant computations – Protect significant computations through algorithm/architecture co-design – Only less-significant computations are affected in case of errors	– Low area overhead – Significant power savings – Need to identify application specific characteristics
ANT	– Estimate potential errors – Detect errors – Select the output closest to the expected	– Significant Power Savings – Area Overhead – Need to design application specific estimators
ERSA	Multi-core systems composed of: – Super reliable core(s) that compute control-intensive data – Less reliable cores that compute computation intensive data	– Reduced overhead compared to redundancy – Higher cost in case of application to kernels other than RMS
SCE	– Utilize techniques in various levels of design abstraction for power reduction and variation tolerance	– Significant power savings – Need to be evaluated in other multimedia kernels
Stochastic processors	– Processors able to select the functional units to be used depending on quality and power requirements	– Power savings – Area overhead – Too many choices – Herqules can help
SSNoC	– Alternative computational paradigm – Network of sensors, the output of which are combined using statistical signal processing techniques	– Power savings – Applicability to generalized computations need to be investigated – Overhead of fusion blocks
Variation-aware power management	– Analysis and application of voltage levels depending on parametric variations and on-line throughput requirements – Application of different voltages to various frequency/voltage islands in case of variable throughput under variations	– Power savings – Robust operation – Extra analysis is needed at design and run time through effective models

reliable RRC hardware cores. In addition, process-aware power analysis and management techniques could be utilized in order to assign the right voltage that will ensure acceptable output quality under various degrees of variations. Herqules, a cross-layer system level design exploration technique, can assist in exploring and identifying the most energy efficient techniques (arithmetic units, algorithmic modifications, voltage, sizing, etc.) at various levels of design abstraction that lead to optimal energy-quality budget for each block and the overall system.

References

1. Abdallah RA, Shanbhag NR (Dec 2009) Error-resilient low-power viterbi decoder architectures. IEEE Trans Sig Process 57:4906–4917
2. Austin T, Bertacco V, Blaauw D, Mudge T (2005) Opportunities and challenges for better than worst case design. In: IEEE Asia South Pacific design automation conference, Shanghai, 2–7
3. Bahar RI, Mundy J, Chen J (2003) A probabilistic-based design methodology for nanoscale computation. In: IEEE/ACM international conference on computer aided-design, San Jose, CA, 480–486
4. Banerjee N, Karakonstantis G, Choi JH, C Chacrabarti, Roy K (Aug 2009) Design methodology for low power dissipation and parametric robustness through output quality modulation: application to color interpolation filtering. IEEE Trans Comput-Aided Des Integr Circuit Syst 28(8):1127–1137
5. Bansal N, Lahiri K, Raghunathan A, Chakradhar ST, (Jan 2005) Power monitors: a framework for system level power estimation using heterogeneous power models. In: IEEE international conference on VLSI design, Kolkata, 579–585
6. Bhavnagarwala A, Tang X, Meindl JD (Apr 2001) The impact of intrinsic device fluctuations on CMOS SRAM cell stability. IEEE J Solid-State Circuits 36(4):658–665
7. Borkar S (Nov 2005) Designing reliable systems from unreliable components: The challenges of transistor variability and degradation. IEEE Micro 25:10–16
8. Borkar S, Karnik T, Narendra S, Tschanz J, Keshavarzi A, De V (Jun 2003) Parameter variations and impact on circuits and microarchitecture. In: IEEE design automation conference, Anaheim, CA, pp 338–342
9. Bowman KA, Tschanz J, Kim NS, Lee J, Wilkerson C, Lu S-L, Karnik T, De V (2008) Energy-efficient and metastability-immune timing-error detection and recovery circuits for dynamic variation tolerance. In: IEEE international conference on integrated circuit design and technology, San Francisco, CA, pp 155–158
10. Bowman KA, Duvall SG, Meindl JD (Feb 2002) Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale Integration. IEEE J Solid-State Circuits 37(2):183–190
11. Breuer MA (Oct 1967) Adaptive computers. J Inf Control 11:402–422
12. Breuer MA (2005) Multi-media Applications and Imprecise Computation. In: IEEE digital system design conference, Lille
13. Chandra S, Lahiri K, Raghunathan A, Dey S (2009) Variation tolerant dynamic power management at the system level. IEEE Trans VLSI Syst 17(9):1220–1232
14. Chandrakasan A, Broadersen RW (1998) Low power CMOS design.Wiley-IEEE
15. Chang H, Sapatnekar SS (Nov 2003) Statistical timing analysis considering spatial correlations using a single PERT-like traversal. In: IEEE international conferenc on computer aided design, San Jose, CA, pp 621–625
16. J I.Chang, Mohapatra D, Roy K (2009) A voltage-scalable & process variation resilient hybrid SRAM architecture for MPEG-4 video processors.In: IEEE design automation conference, San Jose, CA, pp 670–675
17. Chen T, Naffziger S (Oct 2003) Comparison of adaptive body bias (ABB) and adaptive supply voltage (ASV) for improving delay and leakage under the presence of process variation. IEEE Trans VLSI Syst 11:888–899
18. Chippa VK, Mohapatra D, Raghunathan A, Roy K, Chakradhar ST (2010) Scalable effort hardware design: exploiting algorithmic resilience for energy efficiency. In: IEEE design automation conference, Anaheim, CA
19. Cho JH, Schlessman J, Wolf W, Mukhopadhyay S (2009) Accuracy-aware SRAM: a reconfigurable low power SRAM architecture for mobile multimedia applications.In: IEEE Asia South Pacific design automation conference, Yokohama
20. Choi JH, Banerjee N, Roy K (2009) Variation-aware low-power synthesis methodology for fixed-point FIR filters. IEEE Trans CAD Integr Circ Syst 28:87–97

21. Chong E, Zak SH (2008) An introduction to optimization, 2nd edn Wiley-Interscience, New York, NY
22. Dao HQ, Zeydel BR, Oklobdzija VG (2006) Energy optimization of pipelined digital systems using circuit sizing and supply scaling. IEEE Trans VLSI Syst 14(9):122–134
23. De Man H (2005) Ambient intelligence: gigascale dreams and nanoscale realities. In: IEEE international solid-state circuits conference, San Francisco, CA, pp 29–35
24. Ernst D, Kim NS, Das S, Pant S, Rao R, Pham T, Ziesler C, Blaauw D, Austin T, Flautner K, Mudge T (Nov 2004) Razor: circuit-level correction of timing errors for low-power operation. IEEE Microw, 10–20
25. Garg S, Marculescu D (Sep 2008) System-level process variation driven throughput analysis for single and multiple voltage-frequency island designs.ACM Trans Des Autom Electron Syst 13(4):59–84
26. George J, Marr B, Akgul BES, Palem K (2006) Probabilistic arithmetic and energy efficient embedded signal processing. In: Proceedings of the IEEE/ACM international conference on compilers, architecture, and synthesis for embedded systems, Seoul, pp 158–168
27. Ghosh S, Bhunia S, Roy K (Nov 2007) CRISTA: a new paradigm for low-power, variation-tolerant, and adaptive circuit synthesis using critical path isolation. IEEE Trans Comput Aided Des Integr Circ Syst 26:1947–1956
28. Hegde R, Shanbhag NR (Dec 2001) Soft digital signal processing. IEEE Trans VLSI Syst 9:813–823
29. Hughes CJ, Srinivasan J, Adve SV (Dec 2001) Saving energy with architectural and frequency adaptations for multimedia applications. IEEE Micro, Austin, TX, 250–261
30. International Technology Roadmap for Semiconductors. http://public.itrs.net/reports.html
31. Jung H, Pedram M (2008) Resilient dynamic power management under uncertainty, In: IEEE/ACM design automation and test in Europe
32. Karakonstantis G, Mohapatra D, Roy K (2009) System level DSP synthesis using voltage overscaling, unequal error protection & adaptive quality tuning. In: IEEE workshop on signal processing systems, Tampere, Finland
33. Karakonstantis G, Panagopoulos G, Roy K (2010) HERQULES: system level cross-layer design exploration for efficient energy-quality trade-offs. In: IEEE international symposium on low power electronics and design, Austin, TX
34. Karakonstantis G, Banerjee N, Roy K (Nov 2009) Process-variation resilient & voltage scalable DCT architecture for robust low-power computing. IEEE Trans VLSI Syst 99:1–11
35. Kim CH, Roy K, Hsu S, Krishnamurthy R, Borkar S (Jun 2006) A process variation compensating technique with an on-die leakage current sensor for nanometer scale dynamic circuits. IEEE Trans VLSI Syst 14:646–649
36. Koren I, Krishna CM (2007) Fault-tolerant systems. Morgan Kaufmann, San Francisco, CA
37. Kumar A, Rabaey JM, Ramchandran K (2009) SRAM supply voltage scaling: a reliability perspective. In: IEEE International Symposium on Quality Electronic Design, Sauta Clara, CA, pp 782–787
38. Kurdahi FJ, Eltawil A, Yi K, Cheng S, Khajeh A (2010) Low-power multimedia system design by aggressive voltage scaling. IEEE Trans VLSI Syst 18:852–856
39. Leem L, Cho H, Bau J, Jacobson Q, Mitra S (Mar 2010) ERSA: error-resilient system architecture for probabilistic applications. In: IEEE/ACM design automation and test in Europe, Dresden, 1560–1565
40. Marcovic D, Stojanović V Nikolić B, Horowitz MA, Brodersen RW (2004) Methods for true energy-performance optimization. IEEE J Solid-State Circuits 1282–1293
41. Mohapatra D, Karakonstantis G, Roy K (2009) Significance driven computation: a voltage-scalable, variation-aware, quality-tuning motion estimator. In: IEEE International Symposium on Low Power Electronics and Design, San Francisco, CA, pp 195–200
42. Mukhopadhyay S, Kang K, Mahmoodi H, Roy K (Nov. 2005) Reliable and self-repairing SRAM in nano-scale technologies using leakage and delay monitoring. In: IEEE international test conference, Austin, TX, pp 1135–1144

43. Narayanan S, Sartori J, Kumar R, Jones DL (Mar 2010) Scalable stochastic processors. In: IEEE/ACM design automation and test in Europe
44. Palem KV (2003) Energy aware algorithm design via probabilistic computing: from algorithms and models to Moore's law and novel (semiconductor) devices. In: ACM proceedings of international conference on compilers, Architecture and Synthesis for Embedded Systems, Grenoble, pp 113–116
45. Palem KV (2005) Energy aware computing through probabilistic switching: a study of limits. IEEE Trans Comput, 1123–1137
46. Palem KV, Chakrapani LN, Kedem ZM, Avinash L, Muntimadugu KK (2009) Sustaining moore's law in embedded computing through probabilistic and approximate design: retrospects and prospects. In: ACM proceedings of international conference on compilers, architecture and synthesis for embedded systems, Grenoble, pp 1–10
47. Pedram M, Rabaey JM (2002) Power aware design methodologies. Springer, Boston, MA
48. Rabaey (2009) Low power design essentials. Springer, New York, NY
49. Rosing TS, Mihic K, De Micheli G (2007) Power and reliability management of SOCs. IEEE Trans VLSI Syst 15(4):391–401
50. Roy K, Prasad S (2000) Low-power CMOS VLSI circuit design. Wiley
51. Sakurai T, Newton AR (1990) Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas. IEEE J Solid-State Circuits 25(2):584–594
52. Shanbbhag NR, Mitra S, De Veciana G, Orshansky M, Marculescu R, Roychowdhury J, Jones D, Rabaey JM (2008) The search for alternative computational paradigms. In: IEEE Design and Test of Computers 25(4):334–343
53. Shearer F (2010) Power management in mobile devices. Elsevier
54. Shim B, Sridhara SR (2004) Reliable low-power digital signal processing via reduced precision redundancy. IEEE Trans VLSI Syst 12(5):497–510
55. Shrivastava A, Sylvester D, Blaauw D (2005) Statistical Analysis and Optimization for VLSI: Timing and Power. Springer
56. Sinha A, Wang A, Chandrakasan A (2002) Energy scalable system design. IEEE Trans VLSI Syst 10(2):135–145
57. Srivastava A. Sylvester D. (Nov 2004) A general framework for probabilistic low-power design space exploration considering process variation. In: IEEE international conference of computer aided design, San Jose, CA, pp 808–813
58. Tschanz J, Bowman K, De V (2005) Variation-tolerant circuits: circuit solutions and techniques. In: IEEE design automation conference, anaheim, CA, pp 762–763
59. Varatkar G, Shanbhag NR (Oct 2008) Error-resilient motion estimation architecture. IEEE Trans VLSI Syst 16:1399–1412
60. Varatkar GV, Narayanan S, Shanbhag NR, Jones DL (2008) Variation tolerant, low-power PN-code acquisition using stochastic sensor NOC. In: IEEE ISCAS, Seattle, WA
61. von Neumann J (1956) Probabilistic logic and synthesis of reliable organisms form unreliable components. Automata Studies, pp 43–98
62. Wang L, Shanbhag NR (Feb 2003) Low-power filtering via adaptive error cancellation. IEEE Trans Sig Process 51:575–583
63. Wang L, Nicopoulos C, Wu X, Xie Y, Vijaykrishnan N (2007) Variation-aware task allocation and scheduling for MPSoC. In: IEEE international conference on computer aided design
64. Yeo K-S, Roy K (2005) Low Voltage, Low Power VLSI subsystems. McGraw Hill
65. Yuan W, Nahrstedt K (Oct 2003) Energy-efficient soft real-time CPU scheduling for mobile multimedia systems. In: Proceedings of ACM symposium on operating systems principles
66. Zyuban V, Strenski P (2002) Unified methodology for resolving power-performance tradeoffs at the microarchitectural and circuit levels. In: IEEE international symposium on low power electronics and design

Chapter 9
Low-Power Adaptive Mixed Signal/RF Circuits and Systems and Self-Healing Solutions

Shreyas Sen, Vishwanath Natarajan, and Abhijit Chatterjee

Abstract The explosive growth of portable battery-operated devices has mandated design of low-power circuits and systems to prolong battery life. These devices are being designed in modern nanoscale CMOS technologies to allow integration of mega functionalities per chip. The reduced controllability of the fabrication process at these nano dimensions requires the design of process variation tolerant components and systems. This calls for integrated low-power and process-tolerant design techniques, or systems that can adapt to its process and environment to maintain its performance while minimizing power consumption. This chapter provides an overview of design of such Adaptive Low-Power and/or Process-Tolerant Mixed Signal/RF circuits and systems.

9.1 Introduction

The number of transistors in an Integrated Circuit (IC) has been doubling every 18 months following Moore's Law for last few decades. This has allowed the sustained growth of the IC industry by increasing the functionality per chip in every generation. However, increasing functionality per unit area has increased the power consumption per unit area (power density) significantly. Figure 9.1 presents the data by Intel published in early 2000s showing the continued increase in power density. If low-power techniques are not employed the power density of the microprocessors would reach that of a nuclear reactor! Hence a significant amount of research in VLSI in the first decade of twenty first century has been targeted toward low-power IC designs.

The doubling of transistor per unit area has been achieved by technology scaling, i.e., reducing the dimensions of the transistors. Scaling reduces the area as well

S. Sen (✉)
Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA
e-mail: shreyas.sen@gatech.edu

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_9,

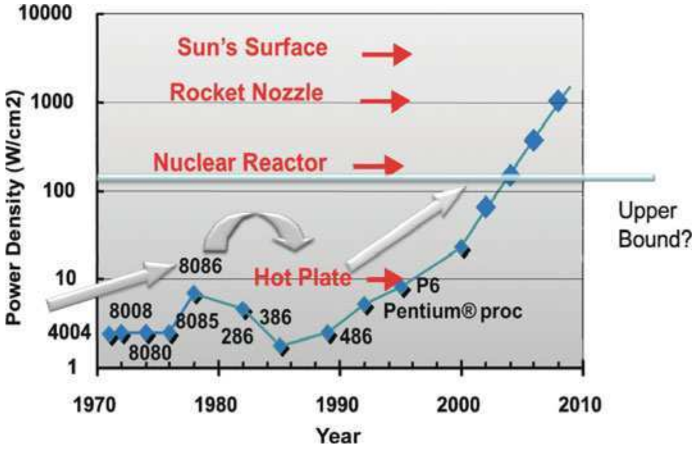


Fig. 9.1 Power density increase in Intel microprocessors over the years [1]

as the capacitance associated with each node making the circuits faster. However, with continued scaling the IC industry is past the sub-micron dimensions and has reached the nanometer regime. The controllability of the fabrication process has reduced significantly due to these nanoscale dimensions of the transistors. This causes variability in fabrication resulting in undesired variation in the circuit and system performance. The traditional static circuits fail to meet the demand of the stringent system requirements due to the underlying variation of the transistors. This calls for a new paradigm of variability aware circuit and system design, where the circuits/systems are intelligent enough to understand the underlying variation and able to heal itself by compensating appropriately (self-healing circuits/systems).

The low power and variation tolerance requirement necessitates new design methodologies as discussed throughout this book for digital circuits and systems. Low-Power and Variation-tolerant Analog/Mixed Signal/RF circuits' require a new/different set of design techniques. In this chapter we highlight the component level as well as system level design challenges in a low-power, process-tolerant Mixed Signal/RF System using a Wireless Transceiver as a demonstration vehicle.

9.2 System Description and Current Demands

9.2.1 Overview of Mobile Transceiver

Figure 9.2 shows a simplified block diagram of a wireless radio transceiver system. A digital baseband DSP sends data to be transmitted through the Digital to Analog Converter (DAC) to the Analog/RF portion of the transmitter. An upconversion mixer up converts the low frequency data into Radio Frequency (RF) signals for transmission. A Power Amplifier (PA) boosts up the RF signal power and transmits

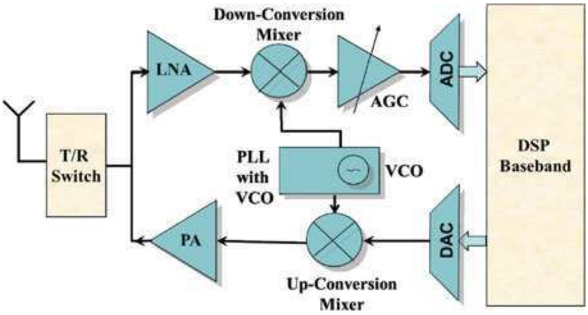


Fig. 9.2 Mobile radio transmitter and receiver

it through the antenna. The Transmit Receive Switch (T/R Switch) selects signals to be transmitted from the transmitter or to be received and sent to the receiver. In receive mode the weak received signal is amplified by the Low Noise Amplifier (LNA) and then down converted to lower frequency by the down conversion mixer. It is then sampled by the Analog to Digital Converter (ADC) and sent to baseband for further processing. A Phase Locked Loop (PLL) including a Voltage Controlled Oscillator (VCO) provides the mixers with required Local Oscillator signals.

The above described architecture is called Direct Conversion Receiver as the low frequency data is directly upconverted to RF frequencies without any intermediate stage. It is very commonly used in wireless radios and would be for demonstrating the concepts of low power and process tolerance throughout this chapter.

9.2.2 Need for Tunability for Low power and Process Tolerance in Wireless Systems

9.2.2.1 Low Power Requirements in Wireless/Portable Systems

With the projected explosion in the number of multimedia wireless applications in the consumer electronics market, power consumption will be a critical metric in future designs of multi-mode multi-standard wireless devices, as most of these devices are portable (battery operated) and battery lifetime is a key demand among consumers. At the physical layer level, low-power design methodologies for digital systems have been studied extensively but relatively fewer techniques have been proposed for Mixed Signal (MS) and Radio Frequency (RF) systems. However a lot of the Mixed Signal/ RFcircuits (e.g., an Analog to Digital Converter (ADC) and a Power Amplifier (PA)) are power hungry and are generally have to be overdesigned to work across different operating environments, process, and temperature. Due to the sensitive nature of Analog circuits and variety of specifications that needs to be satisfied simultaneously, the low-power techniques for analog require different approaches than that of digital as descried here.

9.2.2.2 Process Variation in Mixed Signal Circuits and Yield Loss

With aggressive scaling of CMOS the controllability of the fabrication process is decreasing with each technology node. In the nanometer design regime, high performance Analog/RF circuits are expected to be increasingly susceptible to process variations. It is becoming increasingly impossible to keep the 3σ variations of the process parameters within 10% bound [2, 1]. The variability in the nanometer regime is making circuits less reliable and resulting in significant yield (% of non-faulty chips) loss.

Modern wireless circuits are designed to be extremely high performance as demanded by the consumer. This requires designing the circuits at the *edge*, making the circuits very sensitive to any process variation. In the nanometer regime, the increased variation in the process causes these high-performance systems to vary a lot from its desired specifications, resulting in throwing away of the device. This causes yield loss and results in reduced profit for the IC design company. With static design techniques, if we try to address this issue by leaving an increased guard band, the power consumption goes up. Hence there is a pressing need for variation tolerant circuits which are also low power. Significant research work has been done in recent years in variability aware design of digital circuits. Important techniques include adaptive body biasing [4], dynamic supply voltage scaling [5, 6], and variable size keepers [7]. Radio Frequency (RF) circuits are very sensitive to the increasing process variation. To make RF design successful in the nanometer regime with acceptable yield there is an increasing demand for variability-tolerant RF circuit design techniques, similar to their digital counterpart.

One way to address this issue is to have intelligent circuits and systems that can understand the process corner it is in and tune itself to meet the required specifications, while minimizing power consumption. This requires built-in tunability in Analog/Mixed Signal/ RFcircuits which is traditionally not that common due to the sensitive nature of these circuits.

9.2.2.3 Tunability in Analog/Mixed Signal and RF Circuits

The built in control knobs for required tunability in Analog/RF circuits vary significantly depending on the circuit and voltage-based controls are normally difficult unlike its digital counterpart. This chapter first surveys what are possible knobs for tunable intelligent Mixed Signal/RF circuits. Three key circuits, namely a Low Noise Amplifier (LNA), a Power Amplifier (PA), and an Analog to Digital Converter (ADC) are described here. The possible control knobs and their effects on the above-mentioned circuits are summarized. These tuning knobs could be used in component level as well as system level to address low power and process tolerance requirements. First, examples of *component level* low-power or process-tolerant solutions using these tuning knobs (Section 9.3) are described. Next, a few *system level solutions* for process variation tolerance (Self healing), low power (VIZOR), and low power under process variation (Pro-VIZOR) are described in Section 9.4.

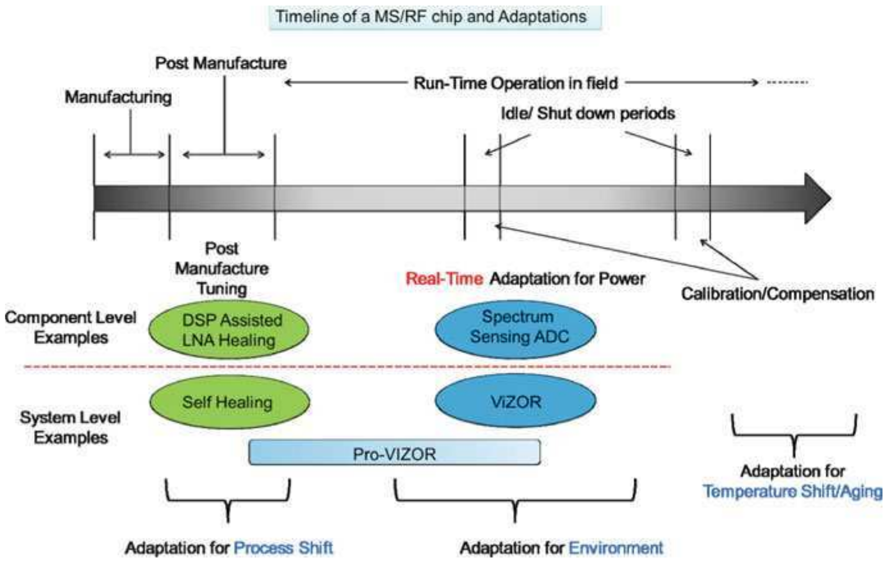


Fig. 9.3 MS/RF chip adaptation timeline

The life-cycle of an MS/RF chip goes through three distinct phases (Fig. 9.3). During the *manufacturing* phase the chip goes through process variation which causes the actual specifications to be off from the desired/nominal values. During *post-manufacture test/tune* the devices are tested for specification compliance and could be tuned for the effect of process variation. The optimal tuning knob settings for each device is found and programmed onto the device as it is sent to be used for real-time operation in field. This can be done in the component level (e.g., DSP Assisted Healing of LNA, Section 9.3) or in the system level (e.g., Power Conscious Self Healing, Section 9.4).

Each static system is invariably overdesigned to meet performance under process/temperature variations and aging as well as to combat environmental variations for wireless systems. This leads to more power consumption than needed. In *real-time* MS/RF systems can be adapted to operate on the "edge" with *Virtually Zero Margin* depending on its operating environment. Component Level adaptation examples for low power includes a Spectrum Sending ADC (Section 9.3) that can trade off performance for power depending on the interference to signal ratio. System Level example includes a complete design framework for Virtually Zero Margin RF systems or VIZOR in Section 9.4. These low-power systems should also be tuned in the post manufacture tune phase to ensure close to optimal low-power operation for any given device. Hence Process-tolerant VIZOR (Pro-VIZOR) systems encompass both the post-manufacture tune phase as well as the real-time operation phase in the above timeline.

During the real-time operation, there are idle or shut down periods when the chip is not operating actively. These times can be used to periodically calibrate or compensate the chip for any temperature effects or aging effects in the long term.

9.3 Component-Level Tunability and Adaptation for Low Power and Process Tolerance Using Built in Tuning Knobs

9.3.1 Tunability in a Low Noise Amplifier (LNA)

An LNA is the first active component after the antenna in the wireless receiver signal path (refer Fig. 9.2). Its job is to provide a high gain to the received weak signal while adding as less noise as possible. The key specifications of a LNA includes Gain, Noise Figure (NF), non-linearity (IIP3), input matching (S11), the dynamic range of the input signal it can process, and the bandwidth (BW). For low-power adaptation supply and bias of the main transistors could be changed, but at the cost of increased NF and non-linearity. Also this does not allow tuning of any frequency shift due to process variation of passive components like inductors and capacitors. Tuning of the input match frequency can be achieved using several-tap tunable inductors [1]. Another way to control only the non-linearity of the LNA is to add an extra current stealing path (Inter-Modulation Distortion or IMD sinker) to a traditional LNA design [8]. The gain of an LNA can be stabilized using damped LC loads [9]. Though this takes care of gain variation to some extent it is not able to compensate for noise figure (NF) and input reflection coefficient (S11). Also this technique is specific to tuned amplifiers. The input and output matching is a critical requirement to ensure proper transfer of signal in RF Amplifier. These matching networks can be made process variation insensitive by breaking then from one step to two step and choosing the components values such that the sensitivity to process variation is minimized [10]. A process and temperature variation tolerant Bluetooth RFIC was designed in [11] by using bandgap references, current compensation circuit, and active loads. The possible tuning knobs in LNA and their effects have been summarized in Table 9.1.

Process variation causes varied specifications of the LNA resulting in yield loss of the complete receiver chip. The above mentioned tuning knobs can be used in component level to tune against process variation and make the LNA process tolerant. It can be done in two ways in the component level as follows:

(1) Sense the variation in the LNA and correct for it through built-in tuning knobs using a DSP-based control. This is a one-time/periodic process controlled by the DSP. The process calibration can be performed during post manufacture tuning phase after fabrication.

(2) Design the LNA to be process variation tolerant, i.e., modify the standard LNA design such that under process variation it compensates for the performance loss by itself in an analog nature. This provides a real-time correction mechanism without the help of any DSP.

9.3.1.1 DSP-Assisted Healing of LNA During Post Manufacture Tuning

A *post manufacture self-tuning technique* aims at compensating for multi-parameter variations in an LNA during post manufacturing test phase [12]. The tunable LNA

Table 9.1 Tunability for low power and variation tolerance in a low noise amplifier

Important specifications	Adaptation for low power and variation tolerance		
	Tuning knobs	Advantage	Trade-offs
Gain, IIP3, S11, Noise Figure, Dynamic Range, BW	Supply	IIP3↓, NF↑, power↓	• Efficient variable supply needed • No frequency tuning
	Bias	Gain↓, IIP3↓, NF↑, power↓	• Efficient variable supply needed • No frequency tuning • BW↓
	Matching inductor [1]	Frequency tuning, S11 tuning	• Several tap inductor required • Reduces Q factor
	IMD sinker [8]	IIP3 control	• Extra current overhead
	Damped LC loads [9]	Gain becomes process tolerant	• Does not compensate for S11 or NF • Specific to tuned amplifiers

incorporates a "response feature" detector and "hardware tuning knobs," designed into the RF circuit. The RF device test response to a specially crafted diagnostic test stimulus is logged via the built-in detector and embedded analog-to-digital converter. Analysis and prediction of the optimal tuning knob control values for performance compensation is performed using software running on the baseband DSP processor. As a result, the RF circuit performance can be diagnosed and tuned with minimal assistance from external test equipment. Multiple RF performance parameters can be adjusted simultaneously under tuning knob control. The concepts illustrated here on an RF LNA design can be applied to other RF circuits as well.

In the design phase, design centering [12] is performed to maximize manufacturing yield in such a way that design parameters for a given circuit topology are optimized toward minimizing performance variability with respect to process, voltage, and temperature (PVT) variations. Such techniques are not IC-specific and are geared toward optimizing the yield statistics across large populations of manufactured die given manufacturing process statistics. In contrast, the post manufacture tuning mechanisms are IC-specific. Tests are applied to each manufactured device, and circuit-level "tuning knobs" are activated in response to the test results to force unacceptable performance metrics to move toward acceptable values. The goal of such post-manufacture tuning is to force these devices corresponding to the tails of

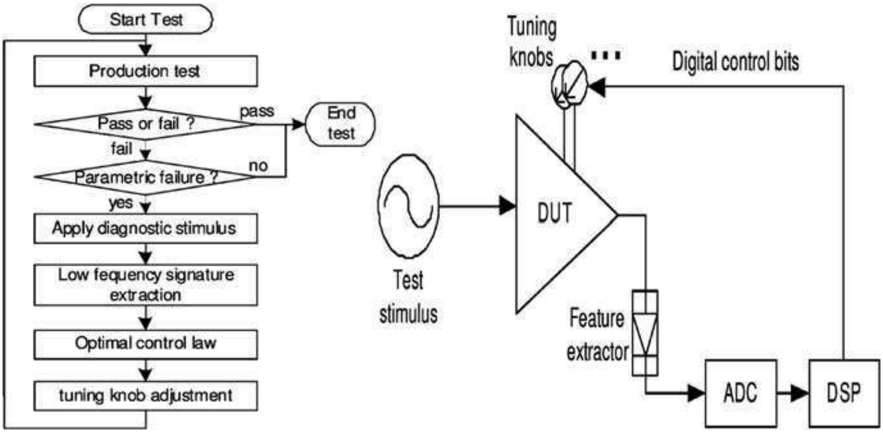


Fig. 9.4 (**a**) Self-tuning procedure; (**b**) Self-tuning hardware [12]

the performance distributions to move toward the mean values of the specifications, thereby improving overall manufacturing yield.

Self-Tuning Framework

The self-tuning method is depicted in Fig. 9.4a and works as follows. After manufacture, production tests are conducted to ensure the proper functionality of a DUT. If the DUT is determined to be faulty with catastrophic faults such as opens or shorts, the procedure terminates. This is because such catastrophic faults are difficult to compensate without replacing the faulty circuitry with redundant units. The self-tuning capability is activated only if it is determined that the current DUT has a *parametric failure* through analysis of test results. The embedded envelope detector is used to convert the test response into a low-frequency test response "signature." This *signature is then mapped onto optimum performance-control tuning "knob" values* via predetermined regression models obtained through a set of experiments performed on the DUTs. The ADC for signature capturing is embedded in a wireless transceiver with a DSP processor. This DSP processor adjusts the digital control bits to trim the tuning control knobs. The self-tuning procedure is performed using the hardware configuration depicted in Fig. 9.4b and consists of the feature extractor (embedded envelope detector), ADC, DSP, and tuning knob modules. The test stimulus employed is designed in such a way that the response envelope extracted by digitizing the output of the envelope detector exhibits strong statistical correlation with the test specifications of the DUT of interest under multi-parameter process variations. In this way, changes in the performance metrics of different DUTs (Gain, IIP3, NF of LNA) are detected by observing changes in the output of the envelope detector for the respective devices.

It is important to note that, in general, the performance specifications of RF/mixed-signal circuits do not change independently of each other under process variations, i.e., changes across different performance metrics are not statistically independent. Consequently, during tuning for process adaptation, each performance metric *cannot be independently tuned,* for a traditional design. Hence, any tuning procedure must consider the impact of tuning on *all* the performance metrics *simultaneously*, since the tunability of one specification may need to be traded off against the tunability of another specification especially when the number of hardware tuning knobs available is limited. Hence, optimal self-tuning involves an iterative optimization process that converges to the optimal values of multiple tuning knobs corresponding to specification values of the DUT as close to the nominal as possible, for different manufacturing process perturbations. This optimization takes into account the interdependence of the several specifications as well as the specification bound requirements set for acceptable yield of a given design. For example, suppose that the requirement for the NF specification of a circuit is stringent, whereas the other specifications have enough performance margins with respect to stated specification limits. In this case, the tuning process focuses mainly on NF compensation. The outcome of this tuning process is a *set of optimal knob settings* for each given process. This is then used to train the regression function which builds a map between the measured signature and the desired tuning knob settings in the training phase. During production test depending on the signature of the test input the optimal control knob settings is applied by the DSP as discussed above to get back the specifications within bound. Next, an application of this methodology on a RF Low-Noise Amplifier with two tuning knobs is described and specification improvements are shown.

GHz CMOS LNA with Two Tuning Knobs

The circuit used for healing should be tunable using some control knobs. Figure 9.5 shows a CMOS LNA with folded p-type MOS (pMOS) IMD sinker [8]. Two tuning knobs were employed to control the bias of the transistors M_1 and M_p. The main stage bias predominantly provides gain controllability whereas the IMD sinker

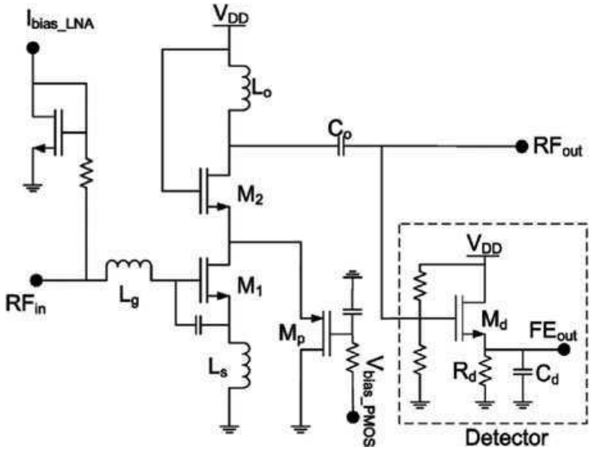


Fig. 9.5 LNA with bias and pMOS IMD sinker as built-in tuning knobs and feature detector [12]

provides IIP3 controllability, though both are not exclusive. Each bias uses a five bit control. Process variation was simulated using Monte Carlo simulations by perturbing the following parameters: zero-bias threshold voltage of p-/n-channel transistors, channel doping concentration, low field mobility of p-/n-channel transistors, and values of passive components such as resistors, capacitors, and inductors. It should be noted that the embedded detector was subject to the same process variations as the rest of the circuitry. For each instance generated via Monte Carlo simulation, all the specifications of interest were measured with the nominal tuning knob values. A two-tone sinusoidal waveform was utilized as the test stimulus (20 dBm at ±5 MHz offset from the center frequency 1.9 GHz). Hence, the fundamental frequency of the envelope response was placed at 10 MHz. The details of the algorithm used to find the *optimal tuning knob* from the observed response could be found in [12]. The specifications of interest were NF, S21, TOI, and Idd. Using the optimal value of the tuning knobs the tuned values of the specifications are achieved. Figure 9.6 shows the distribution of NF and Gain (S21) before and after tuning for the validation set of DUTs. All the performance variabilities are significantly reduced without impacting the mean value of each specification. The perturbations of each specification are listed in Table 9.2 in terms of their mean

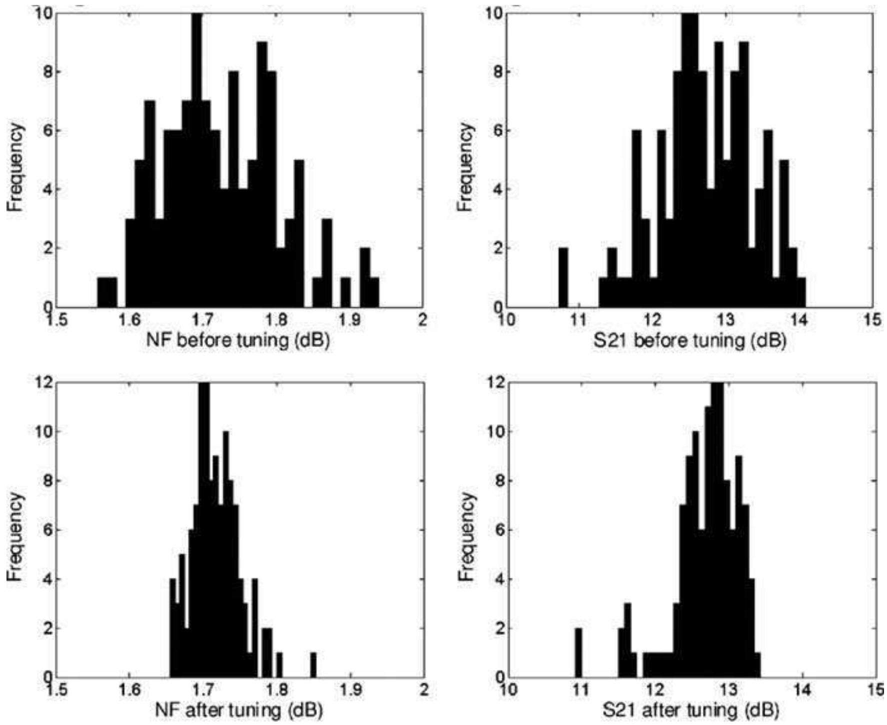


Fig. 9.6 Specification distribution before and after tuning [12]

Table 9.2 Changes of the specification after tuning [12]

	Before		After		
	μ_b	σ_b	μ_a	σ_a	σ_a/σ_b
TOI	15.76 dBm	2.86 dBm	15.77 dBm	0.86 Bm	0.30
NF	1.72 dB	0.07 dB	1.72 dB	0.03 dB	0.43
S21	12.64 dB	0.62 dB	12.68 dB	0.46 dB	0.74
Idd	7.10 mA	0.14 mA	7.09 mA	0.07 mA	0.5

and standard deviation. For example, the standard deviation of the specification TOI shows 2.86 dBm before tuning and is reduced to 0.86 dBm after tuning. It is also observed that standard deviation of the I_{dd} (i.e., power consumption) is reduced without compromising its mean value. Hence, post manufacturing tuning technique provides *healing* of the LNA under process variation.

9.3.1.2 Stand-Alone Self Compensating Process-Tolerant LNA

Another approach for design of process-tolerant RF circuits is to not rely on DSP at all. *Self-compensating process variation tolerant RF Low Noise Amplifier* (*LNA*) can be designed using minimal intrusion negative feedback in RF circuits. The example shown in [13] of this design technique provides 18% yield improvement over comparable conventional LNA under severe process variation by developing a non-intrusive mixing technique as well as minimizing the intrusion of the sensing circuit. This enables use of negative feedback in RF circuits and helps in designing process variation tolerant RF front ends. Figure 9.7 shows the design of process variation tolerant LNA. The analysis of the circuit could be understood by following

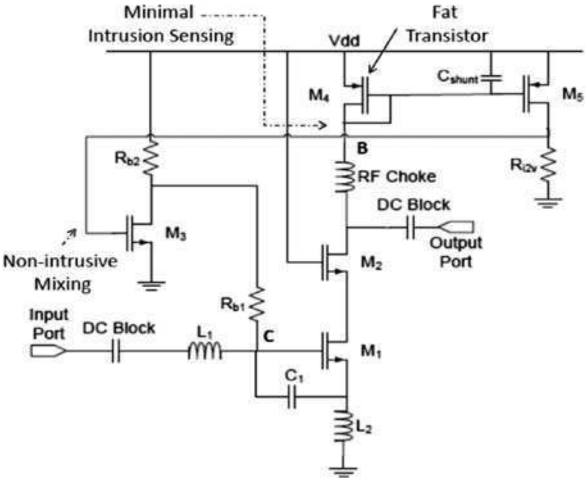


Fig. 9.7 Schematic diagram of process variation tolerant LNA [13]

the feedback loop. Due to process variation, say I_d has increased from its nominal designed value. This results in similar increase in the mirrored current I_{M5} and hence increases the voltage across R_{i2v}. This acts as the gate to source voltage of M_3 ($V_{G,M3}$). As $V_{G,M3}$ goes up drain voltage of M_3 goes down reducing the biasing voltage (point C) of the transconductance stage, in turn reducing I_d. Thus through this negative feedback loop this circuit keeps the I_d constant (as close as possible to the nominal designed value of I_d) or in other words keeps the LNA biased "properly" even under severe process variation. Under RF operation since $Z_{in,B} \rightarrow 0$ it does not have much effect on the circuit and keeps the RF performance of the circuit intact.

Yield Recovery Through Process Variation Tolerant LNA

A conventional LNA and the process variation tolerant LNA, working at 1.8 GHz with similar gain, noise figure (NF) and input return loss (S11) shows the yield improvement using this design. Severe process variations were introduced to important process parameters namely, minimum channel length (L_{eff}), oxide thickness (T_{ox}), threshold voltage (V_t), capacitances (C) and inductances (L).

Yield analysis with the following performance constraint: 11 dB<Gain<17 dB, S11<–10 dB and NF<1.35 dB shows a significant yield increase of 18% for the process-tolerant circuit compared to conventional counterpart. Figure 9.8 shows the distribution of % of ICs based on gain, NF and S11 obtained from yield analysis. It can be seen that for all the specs process-tolerant LNA performs better than the conventional one (less variation around its nominal spec). Hence the above-described techniques allows design a process variation tolerant LNA which self-compensates under variation without any DSP. This technique can very easily be extended to other RF components opening up the paradigm of variability-aware RF front end design.

9.3.2 Tunability in a Power Amplifier (PA)

A PA is the last block in a wireless transmitter that drives the signal to be transmitted to the antenna (refer Fig. 9.2). Due to high-transmission power requirements the PA tends to be the most power hungry block in the transceiver. It needs to provide gain to the signal while delivering high output power, i.e., process signals with extreme large swing (typical swing is close to 1.7–2 times the supply voltage). The key specifications include Gain, Non-linearity (IIP3), input and output matching, Adjacent Channel Power Ratio (ACPR), bandwidth (BW), and efficiency. Since the signal levels in a PA are very high the tunability achieved by gate and drain bias control are minimal. Voltage-based tuning can be aided by dynamically companding in the digital to reshape the signal to reduce its peak to average ratio [14]. Output of the PA has to conform to stringent spectrum requirements set by the Federal Communications Commission (FCC). This requires highly linear operation of the PA, which reduces its efficiency or makes it power hungry. A pre-distortion function, i.e., the inverse

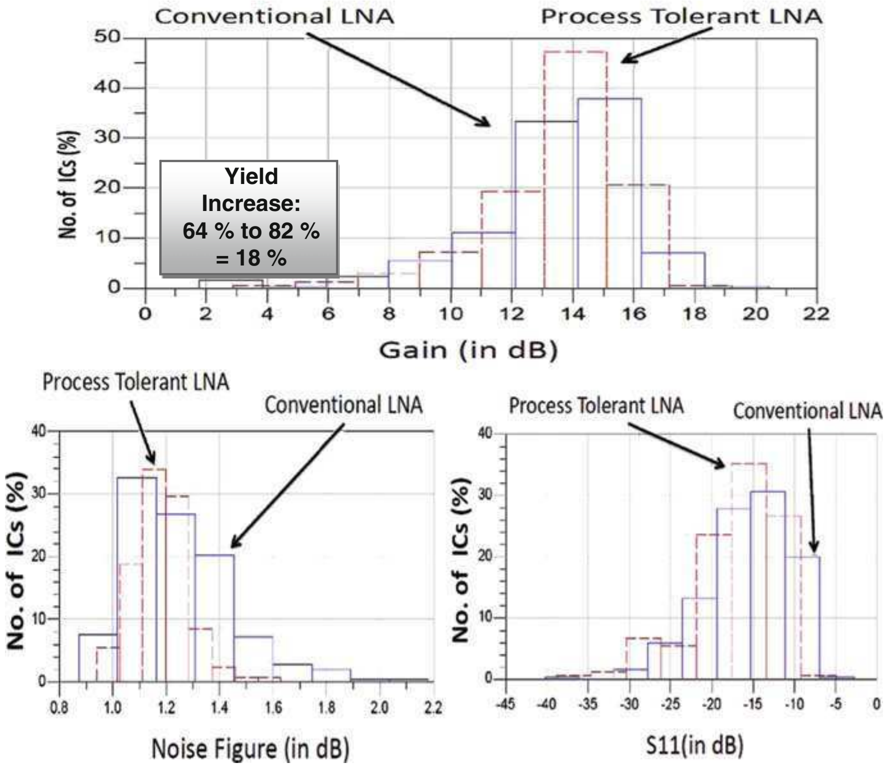


Fig. 9.8 Variation of gain, NF, and S11 of the process variation tolerant LNA and conventional LNA [13]

of the PA nonlinearity is used in the baseband DSP to increase the overall linearity of the system even with low-power consumption.

In the system level, to reduce interference and increase efficiency, *output power* of the Mobile Station (MS) is varied depending on the channel condition and distance from the Base Station (BS). Using power control information (based on received signal strength) sent by the BS to MS through the BCH (Broadcast channel) [1, 15–20], the PA average output power is varied. If the DC power consumption is kept fixed (i.e., static PA) the efficiency of the PA is very low for the time that it is not transmitting at the maximum average output power level. An approach to increase the operational efficiency of RF transmitters has been to use a tunable PA (such as Agilent ACPM-7891 [21]) in the front-end. A MS power controller (such as National Semiconductor LMV243 [22]) uses the information from BS to control the output power of the PA by applying proper control voltages and sometimes using a control loop [20, 23] .Output power is varied in a wide range as described in [24]. For example, output power could be varied from 5 to 39 dBm for GSM 900 MHz band. This recovers some of the efficiency loss due to variable distance between the BS and MS.

To increase the efficiency of PAs using circuit level techniques, the supply voltage of the PA is changed according to the envelope of the signal. These PAs can be broadly divided into three categories [25], namely *slow tracking*, *envelope tracking*, and *polar modulation* PAs. A *slow tracking* PA [26] supplies the PA with a voltage slightly greater than the largest peaks of the envelope. Hence it can only recover the efficiency loss due to power-control back-off. An *envelope following* or *envelope tracking* PA [27–29] have been developed in which the supply voltage is dynamically modulated by the estimated or tracked envelope to keep drain efficiency high. Hence it allows recovery of lost efficiency due to both modulation and power-control back-off. In these methods, the envelope amplitude information has to be extracted from the signal and incurs extra complexity and increased hardware overhead. In *polar-modulated* PAs [30, 31] the supply modulator applies the required envelope signal directly onto the carrier through a saturating PA. Though the use of saturated PAs increase efficiency, the bandwidth requirements of the AM path is significantly high and the noise requirements of the supply modulator makes this implementation impractical. A power management block presented in [32] provides good efficiency while providing variable output voltages. This on-chip block can generate drain and gate bias for a PA using inputs from DSP making adaptation of PAs easier. This work also shows adaptation of a PA with varying average output power to maintain high efficiency, using the above mentioned PMU.

Another significant effect in the PA is input power dependent gain compression and phase distortion. AM–PM distortion can be corrected by introducing an opposite phase shift to cancel the signal power dependent phase shift of the PA. Figure 9.9 shows a schematic that achieves this goal. A varactor is placed across one of the inductors in the mixer or the PA. By changing the control voltage of the varactors [16], the resonance frequency of the tank can be changed, thus modifying the phase shift incurred by the signal. The control voltage changes according to the amplitude of the input signal in such a way as to cancel the AM–PM distortion of the PA. This shows a real-time technique to correct for AM–PM. Under process shift, the AM–PM curve changes for a PA. This can also be compensated for using this technique just by changing the way control voltages are applied.

The possible tuning knobs in a PA and their effects have been summarized in Table 9.3.

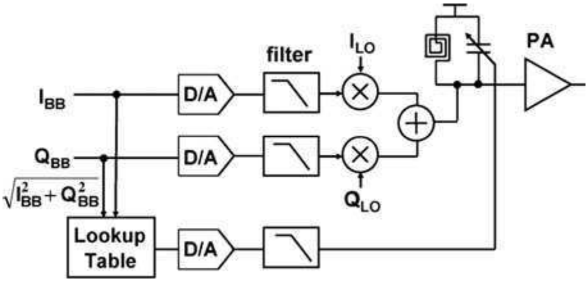


Fig. 9.9 Feed-forward AM–PM correction using varactor tuning [16]

Table 9.3 Tunability for low power and variation tolerance in a power amplifier

Important specifications	Adaptation for low power and variation tolerance		
	Tuning knobs	Advantage	Trade-offs
Gain, P1dB, efficiency, IIP3, ACPR, 3 dB BW matching	Drain bias	P1dB↓, power↓	• Efficient variable supply needed • Non-linearity↑
	Gate bias	Gain↓, P1 dB↓, power↓, efficiency↑ @ low output power	• Efficient variable supply needed • Non-linearity↑
	Dynamic companding [14]	PAR↓→ Efficiency↑	• Channel estimation based feedback required
	Pre-distortion	Non-linearity↓	• Extra digital processing required
	Varactor [16]	AM–PM↓	• Careful matching required

9.3.3 Tunability in an Analog to Digital Converter (ADC)

An ADC is the interface between the Analog/RF and digital section in the receiver chain that converts the analog received and amplified signal to digital for further processing. Similarly, a Digital to Analog Converter (DAC) converts the digital signals to analog in the transmitter (refer Fig. 9.2). Modern communication standards require increasingly higher BW at lower power, making the ADC designs highly challenging. The key specifications of an ADC include the resolution (number of bits), dynamic range, BW, integral non-linearity (INL), differential non-linearity (DNL), signal to noise ratio (SNR), and signal to quantization noise ratio (SQNR). One way to introduce power performance trade-off in ADCs for tuning is to use adaptive step size based on the received signal. Another way to reduce power is to reduce the resolution dynamically by dropping the number of bits. This introduces power vs. SNR trade-off [18].

To reduce power consumption dynamically the order of the loop filter can be reduced dynamically depending on the interference environment in a sigma delta ADC [17]. This could be achieved as shown in Fig. 9.10. The *reconfigurable ADC* consists of several modes allowing power versus performance trade-off. As the interference to signal ratio goes down the requirement of the ADC dynamic range is relaxed and hence a lower power mode can be used. A low resolution flash ADC is used for spectrum sensing. A simple spectrum Analyzer is used for spectrum sensing from the flash ADC output and the mode of the ADC is set based on that. This allows power savings by choosing lesser performance when interference in the received signal is low.

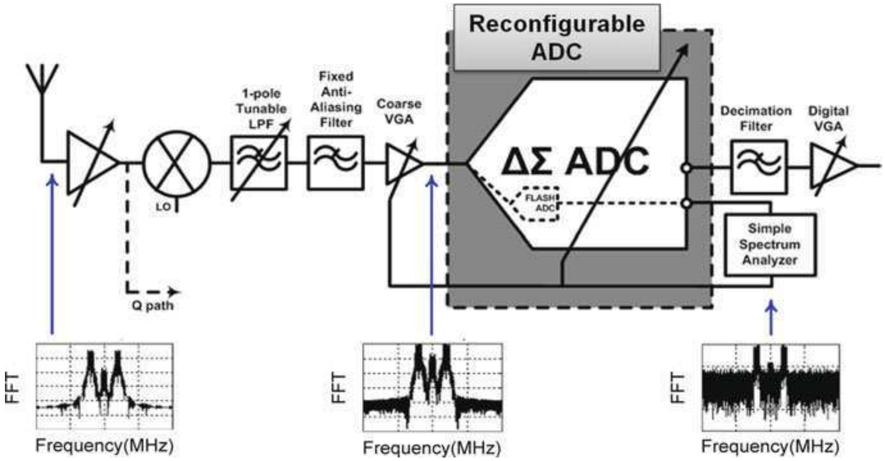


Fig. 9.10 Reconfigurable ADC in a wireless receiver for reduced power consumption when interference is less [17]

Table 9.4 Tunability for low power and variation tolerance in an ADC

Important specifications	Adaptation for low power and variation tolerance		
	Tuning knobs	Advantage	Trade-offs
Resolution, dynamic range, BW, INL, DNL, SNR, SQNR	Adaptive step size	SNR↑, power↓	• Extra hardware • Adaptive feedback loop
	Dynamic loop filter order reduction [17]	power↓	• Adaptive feedback loop
	Dynamic bit drop [18]	Resolution↓, power↓	• Channel estimation based feedback required

The possible tuning knobs in an ADC and their effects have been summarized in Table 9.4.

Similar tuning knobs could be found in other Analog/RF/Mixed Signal blocks, though there are not any standard set of knobs that works across all circuits. To date significant research is going on in identifying efficient tuning knobs in these circuits that enhances adaptability. Until now we have looked at what sort of tuning knobs are available in MS/RF circuits and how they can be used in component level to achieve low power and process variation tolerance. Though several component level techniques possible, a unified system level methodology promises significant power saving and/or yield improvement as it considers the interaction between all the complex specifications that dominate a complete MS/RF system. Next system level techniques for low power and process tolerance of MS/RF systems are discussed showing that significant benefit is achieved when several knobs are tuned simultaneously using a system level framework.

9.4 System Level Approach to Process Variation Tolerance and Low Power in Mixed Signal/RF Circuits and Systems

Until now some of the possible tuning knobs have been highlighted in some important MS/RF components. In this section we will look into how these tuning knobs can be used efficiently in the system level to achieve process variation tolerance and low-power solutions. We start with an example of a real-time adaptation methodology that adapts a wireless transceiver to its operating environment to reduce built in design margins (Virtually Zero Margin RF: VIZOR) on the fly and save power. The operation of these systems under process variation is also described. Next, process variation tolerant *Power Conscious Self-Healing* transceiver systems that tune itself, in the post-manufacture tuning phase for specification compliance with minimum possible power consumption leading to increased yield of the system is described. This is a time t=0 adaptation right after production, while VIZOR is an example of real-time adaptation methodology.

9.4.1 Low-Power Operation of Wireless Systems Under Dynamic Environment

The low power requirements for portable devices start to become even more critical for portable wireless devices. This section provides an overview of the challenges faced by these portable wireless devices and describes traditional solutions and futuristic solutions leading to environment-adaptive wireless systems.

9.4.1.1 Need for Environment-Adaptive Wireless Systems

Most of the modern portable systems include a radio for wireless communication. The key difference between non-wireless portable systems and portable wireless systems is that wireless systems have to transmit and receive through a channel, which keeps on varying. Hence these systems need to operate at highest performance when the environment is at its worst. Static systems would waste a lot of power as they would over perform when the environment (channel) is better than its worst-case condition. One way to increase battery life of these systems is to design low-power circuits and efficient algorithms. But static low-power systems are still non-optimum in terms of efficiency. Power efficient wireless systems can be designed by adapting the system and the underlying circuits as the environment changes, to deliver just the required amount of power.

To make the radio power efficient both transmitter and receiver should have low power implementation. In the transmitter the PA is the most power hungry block. Hence there are several techniques to reduce the PA power consumption, i.e., increasing its efficiency as described in Section 9.3. Reducing the power consumption of the receiver is even more challenging as all the components play a significant role in power dissipation. At the physical layer, low-power design methodologies for digital systems [33, 34] and analog/RF systems [35, 36] have been studied extensively. The issue with these approaches though is that circuit designs incorporate margins to account for *worst-case* estimates of process variability, thermal effects in the RF front-end, and channel conditions corresponding to different modes (data rates) of transmission. However, for most of the time a wireless device is powered up for operation, it *is not in a worst-case environment* and hence, consumes more power than necessary under the majority of operating conditions. The need for dynamically adapting RF front-end, baseband and digital circuits to save power and enable multi-mode operability in future wireless devices was discussed in [37–39]. There exists media access control (MAC) and network-level dynamic power management schemes [40] that conserve power by adapting the data rate (modulation and coding rates) based on certain channel quality metrics derived from the analysis of training symbols. Present-day wireless devices also feature high-power, low-power, and shut-down modes that are activated on the basis of prevailing channel conditions. Though these approaches are effective in reducing the power consumption levels, they do so in a few discrete steps, and hence do not fully exploit the design margins through fine-grained system-level adaptation. An energy scalable RF transmitter is shown in [41], where the front-end is dynamically tuned (supply, bias, and resistances) for each data rate modulation set by the higher-level link layer protocol. The tuning is driven by channel quality as determined by channel estimation metrics and relies on simulation of a large number of channel and RF front-end settings.

The trend in low-power Analog/RF systems is hence twofold. One is to design circuits with inherent low power consumption. The second is to adapt those circuits over process, temperature, environment, workload, etc. This leads us to complete system level adaptation of wireless front ends using end-to-end metrics for ensuring performance. These systems are called Virtually Zero Margin RF (VIZOR) as they thrive to take out the built-in design margins that is a must in static circuits and systems.

9.4.1.2 Components of Environment-Adaptive System

The enabling technology (VIZOR) that operates the wireless device at minimum power consumption levels across all environmental (channel) conditions consists of:

- *Tunable Circuits*: The wireless circuit in an environment-adaptive system should exhibit power versus performance trade-offs using some built-in control knobs as described in Section 9.3. In the example shown here tunable LNA, Mixer, ADC, and PAs are used.

- *Channel Sensing:* The wireless device should be able to accurately estimate the current operating environment so that it can adapt accordingly. Here environment refers to the transmission channel between the wireless access point (AP) or base station (BS) and the mobile station (MS). This channel information is used to drive the adaptation of the transceiver. A simple example of channel sensing is through an adaptation metric called Error Vector Magnitude or EVM as described later.
- *Environment Adaptability:* Given any operating channel, the system should consume only the minimum amount of power required to maintain desired performance. This requires that the components (RF circuits, ADC, and baseband processing) of the wireless device to be dynamically adapted using an *optimal control law,* such that the adaptation is optimal as the channel changes.
- *Control Algorithm for Environmental Adaptation:* A system-level multi-dimensional control algorithm that actuates environment sensing and adapts the device using closed-loop feedback is required. The tuning of system components is based on an optimal control law ("locus") that is obtained during pre-production device characterization phase.

Under process variation this locus might not be optimum anymore, requiring additional steps to maintain the benefits of this VIZOR system. The extra components required in a process-tolerant VIZOR (Pro-VIZOR) system are:

- *Process Sensing:* To ensure optimum adaptation to varying channel conditions, the device should be able to self-test and ascertain the performance sensitivity of the individual circuit components to adaptation. Due to the process spread in manufactured devices, the adaptation in each device should be *calibrated* to enable minimum power operation. Unlike channel sensing (which is continuous), process sensing is a one-time procedure performed during production test.
- *Process Adaptability:* To ensure effective environmental adaptation under process variation, a process adaptation routine drives the process estimation and then using that information find the modified optimum locus for the process skewed device.

The above mentioned components will be described in detail in the following subsection leading to the design of a process-tolerant environment-adaptive low-power system.

9.4.1.3 System Level Adaptation: Virtually Zero Margin RF (VIZOR)

The overall framework for transceiver adaptation is shown in Fig. 9.11. The mobile station (MS) is the point of interest where power consumption is minimized to increase battery lifetime using adaptation in both the transmitter and receiver. A real-time system-level adaptation approach for a tunable wireless transceiver, driven by closed-loop feedback control based on an adaptation metric, is presented here. This *metric* (in this case, illustrated via Error Vector Magnitude (EVM) of the received

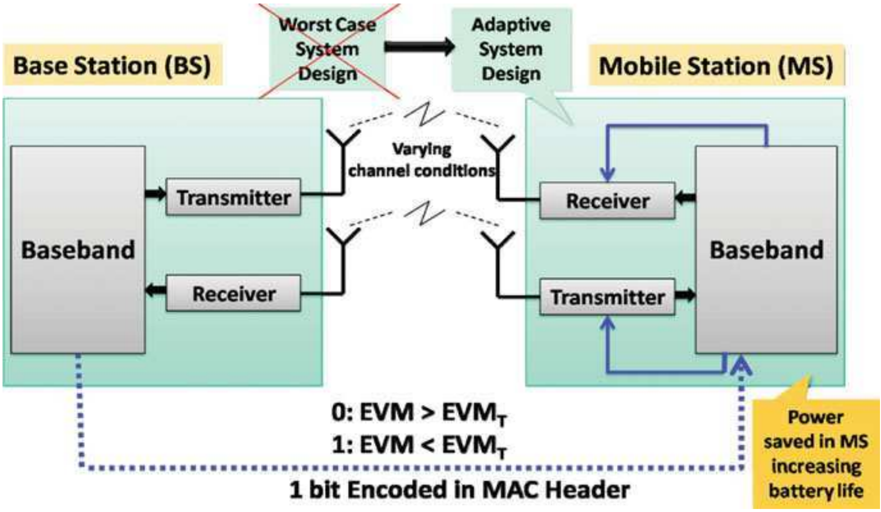


Fig. 9.11 System Level Adaptation Framework [45]

symbols) exhibits strong statistical correlation with Bit Error Rate (BER), i.e., usually used for characterizing the performance of a wireless link. The feedback control of the wireless device is designed so that this EVM value is always close to a specified upper limit irrespective of channel (environment) conditions. The EVM is calculated in the MS receiver baseband, and this information is used to govern the tuning of the receiver. Performance of the receiver is increased if calculated EVM is greater than the threshold EVM (EVM_T) (corresponding to the maximum BER), and vice versa. The EVM always hovers around its threshold value, providing just enough power to the system to maintain desired performance. In contrast the MS transmitter performance is tracked by computing the EVM in the baseband of the BS. To facilitate power versus performance trade-off of the MS transmitter, this EVM information has to be delivered to the MS from the BS. This is achieved by encoding the information into the MAC header of the next data packet transmitted from BS to MS. The assumption here is the environment changes slowly compared to the duration of the packets, which is valid for a majority of the time that the device is in operation. To keep the transmitted MAC bit overhead minimum only 1 bit reporting scheme is adopted. The BS sends back 0 or 1 depending on EVM is greater or lesser than EVM_T respectively. Using this information the MS baseband controls the tuning of the MS transmitter. Both the transmitter and receiver adaptation is performed in a standard compliant manner from data rate perspective, meaning this algorithm works in conjunction with higher level data rate switching protocol and strives to operate at minimum power by operating the system close EVM_T (different for different modulation) for any given data rate. A comprehensive system-level framework tunes multiple system control "knobs" that include RF and digital blocks. A control law running on the baseband processor tunes these knobs while simultaneously performing baseband compensation on the received data to save power across all operating conditions without compromising the BER specification. Thus, by dynamically trading off performance (when not required) such an *Adaptive System* [8, 42–44] shows significant power savings compared to *Worst-Case Systems*.

A Suitable Adaptation Metric: EVM

The key issues to be considered in developing a dynamic feedback-driven power control for wireless systems include defining a suitable adaptation metric and the acceptance bounds on this metric for satisfactory operation. This metric should provide the best indication of the system performance under all possible environmental conditions and should be estimated in run-time. Error Vector Magnitude (EVM) fits the requirements. It quantifies the difference between the transmitted (ideal) and received (distorted) modulated data. The received signal passes through all the receiver components before it is demodulated and decoded in the baseband processor. The EVM specification therefore captures the *cumulative effect* of the environment such as attenuation, interference, multipath fading, etc. as well as all the circuit-level specifications such as gain, non-linearity, noise figure, input/output match, ADC resolution, etc. EVM computation is a byproduct of the normal baseband processing and represents minimal hardware/power overhead. Figure 9.12 shows the relation between EVM and BER for three modulation scheme, namely QPSK, 16-QAM, and 64-QAM. It also shows the variance in this metric for guard band estimation purposes. A good correlation between these metrics and a low variance allows using EVM as the adaptation metric. For example, if BER bound is set at 1e^{-3}, the corresponding mean EVM bound for the QPSK and 16-QAM cases can be approximated to about 35 and 12.5%, respectively.

9.4.1.4 VIZOR Receiver

Adaptive Receiver Design Framework

Figure 9.13 shows the components of the Adaptive Receiver. It consists of both tunable RF components (LNA and mixer) as well as a tunable mixed signal block (ADC). The tunable supply/bias voltages for LNA and mixer, along with the tunable analog-to-digital converter (ADC) word length serve as control "knobs" for EVM-based feedback control. These tuning knobs provide a way to reduce performance for reduced power operation of the above mentioned blocks. The transmitted signal from the tower goes through a varying channel and passes through the receiver and gets sampled by the ADC. In the DSP, the sampled signal is compensated for known (pre-characterized) non-ideality of the current front end configuration and passed though baseband OFDM processor, which calculates EVM as a byproduct. If the EVM is less than the EVM threshold for the current modulation the power control

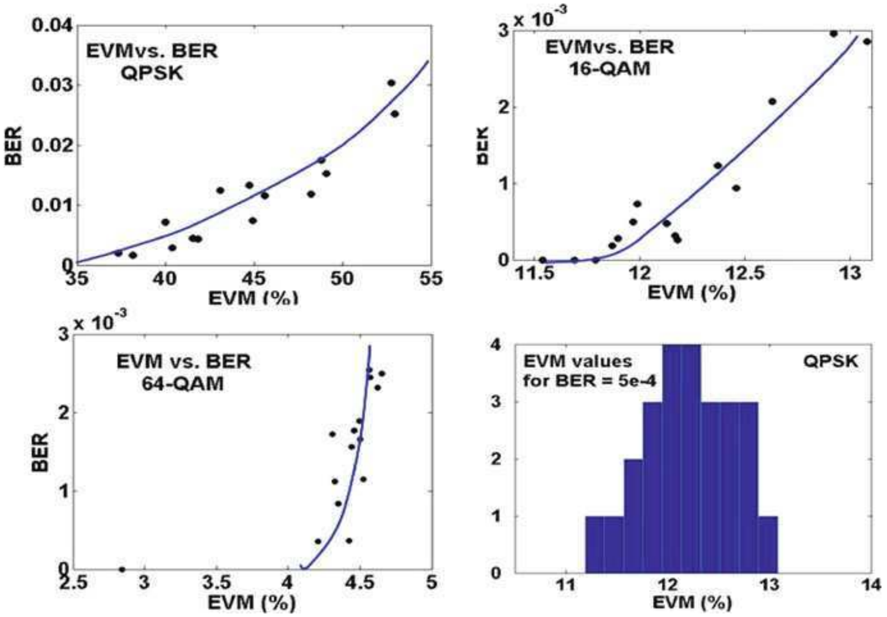


Fig. 9.12 EVM vs. BER relations for QPSK, 16-QAM, 64-QAM and EVM guard band estimation for QPSK

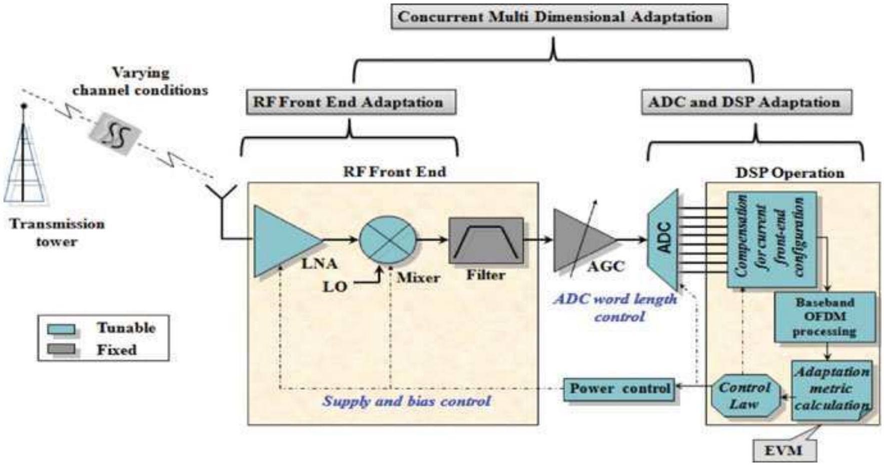


Fig. 9.13 Adaptive Receiver Framework [18]

block reduces the tuning knobs of the tunable components following a predefined *control law* to save power. This control law defines how all the “knobs” change in relation to each other when the receiver performance is being turned down. This calls for a multi-dimensional tuning algorithm that optimizes how the knobs should change (i.e.,the *control law*) in design phase and applies it to the adaptive receiver in run time to guarantee optimum performance of a nominal adaptive receiver.

Design Phase Optimization

During the design phase, a multi-dimensional optimization algorithm is used to determine a *Minimum Power and Maximum EVM locus* (that defines how to turn down the "knobs" with respect to each other). A set of channel conditions ranging from good to bad are modeled for use in the optimization procedure. The channel parameters (interference, multipath, and attenuation) are perturbed to obtain channels ranging from good (low EVM) to bad (high EVM). For each of these channel conditions, the optimal settings for control "knobs" are computed through a multi-dimensional optimization procedure as described below. Each optimized point refers to a set of knob values that provide the optimum power for the given channel and is a point on the locus (refer Fig. 9.14). The set of all points for all the channels define the optimum *control law*.

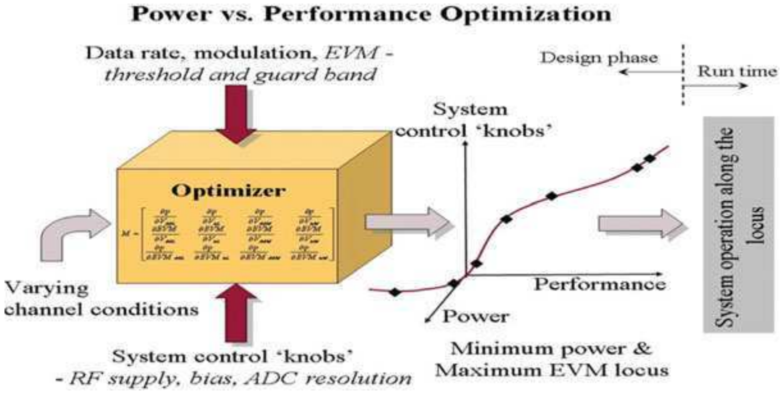

Power vs. Performance Optimization
Data rate, modulation, EVM - threshold and guard band
Design phase
Run time
System control 'knobs'
Optimizer
Varying channel conditions
Performance
Power
System operation along the locus
System control 'knobs' - RF supply, bias, ADC resolution
Minimum power & Maximum EVM locus

Fig. 9.14 Receiver power optimization [45]

Development Optimal Control Law

The constraints on the optimizer is summarized as

Input: Maximum allowed EVM for each signal modulation (QPSK, 16-QAM, and 64-QAM).

Goal: For each signal modulation, find V_{dd}, V_{bias}, W tuning $= f(\text{channel quality})$ (channel quality) such that power (P) is minimized and EVM = EVM threshold – EVM guard band.

The implementation details are outlined below.

1. For a given channel, starting from the nominal set of values for the supply and bias voltages and ADC word size, the following adaptation vector (A) is calculated

   $$A = \left[\frac{\partial P}{\partial EVM_{\text{Vdd}}} \frac{\partial P}{\partial EVM_{\text{VBias}}} \cdots \frac{\partial P}{\partial EVM_{\text{W}}} \right]$$

   Each entity in A is the ration of change in power consumption of the device and EVM change for a unit change in the corresponding control knob setting (given by the suffix).
2. For every iteration during the optimization procedure, the control knob (V_{dd}, V_{bias}, or W) corresponding to the maximum of the four entities in the third row is selected. This allows us to tune the knob that causes the maximum reduction in power consumption for the least increase in EVM.
3. Once a particular voltage parameter is selected, it is scaled down to generate a new set and A is recomputed, and the procedure is repeated.
4. For each channel, the iteration steps continue until the EVM threshold condition is violated or all possibilities are exhausted.
5. A table of these voltage values obtained across a range of channel conditions defines a *optimal locus* of points in an N-dimensional space for N available tuning knobs. The control law moves the tuning knobs only along this locus for minimum power operation during run time.

Run-Time Operation of the Device

During run time, the control law forces continuous adaptation of the device to minimize power while meeting the BER constraint at all data rates and channel conditions. This is done by operating the system along the optimal locus of the control knob settings obtained from the optimization procedure. The threshold and guard band for EVM is set based on the estimated channel quality and current data rate. The voltage scaling is performed independently for each modulation in a standard-compliant manner such that the received signal quality meets the required bit error rate specification for each modulation. The run-time operation is shown in Fig. 9.15a. EVM keeps on increasing as the VIZOR controller throttles the receiver power lower and lower until EVM threshold is reached. Figure 9.15b shows the power of the system as it adapts, highlighting the power savings in lower power bias points than the highest (nominal) one.

The design of the power control circuitry is crucial in this approach, as it determines the power savings. Low-power operability would be limited by the response and settling time of the feedback control circuitry. Careful optimization of the control loop parameters is necessary to ensure stability, especially under fast varying channel conditions. Response times of the order of a few micro seconds for power management blocks have been presented in literature [32], whereas the channel variation is much slower compared to this, allowing this method to be feasible. In case

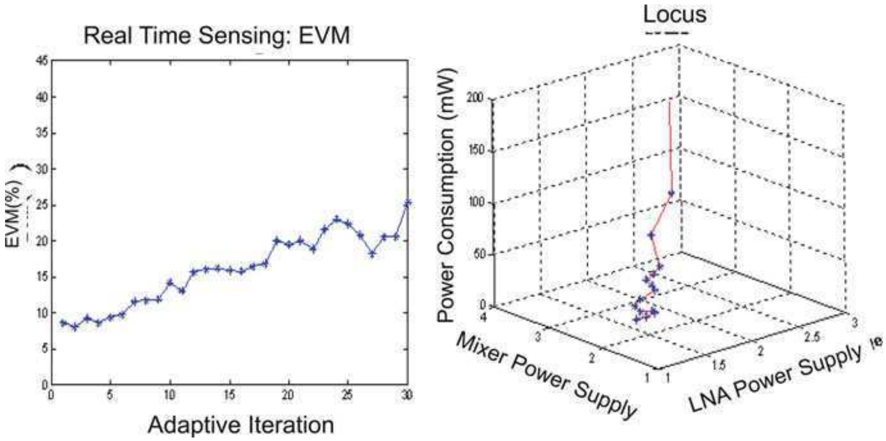


Fig. 9.15 (**a**) EVM variation with time and (**b**) operation along the locus showing power savings compared to the static case (highest point on the locus) [45]

the channel changes abruptly the VIZOR controller takes the system to full power operation to avoid significant data loss.

Power Savings

The power-EVM optimizer is used to obtain the locus of optimal control knob settings for a set of 12 different channels (locus points) ranging from good to bad. Three popular modulation schemes – QPSK, 16-QAM, and 64-QAM are studied. The optimal RF front-end power consumptions across these channel conditions are plotted in Fig. 9.16a for the three different modulations. It is observed that for a majority of the channel conditions, the optimal RF power consumption is lower than nominal value of 48 mW. It is also observed that the optimal control knob settings and the associated power consumption is lowest for QPSK modulation. This

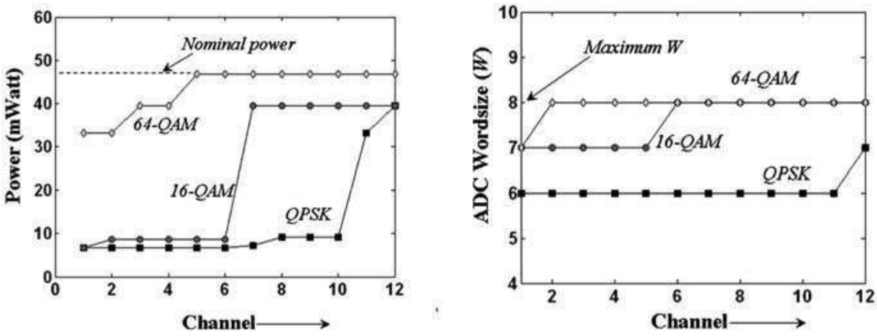


Fig. 9.16 (**a**) Power consumption for different modulations and (**b**) optimal ADC wordsize (*W*) along the locus for different modulations [45]

is due to the tighter requirements on the SNR for 16-QAM and 64-QAM (higher data rates). The optimal *W* required along the locus points is shown in Fig. 9.16b. As observed, simulations indicate that sufficient margin exists for pruning *W* (maximum of 8 bits) under favorable channel conditions. Up to 2 bits of resolution can be sacrificed for QPSK modulation under majority of the channel conditions. Though the margin is lower in the case of 16-QAM and 64-QAM modulation, the system budget allows for a bit drop under good channel conditions.

In wireless standards such as WLAN, the higher-level MAC protocol dynamically changes the data rate (modulation and coding) based on the channel conditions. The control law operates within the framework of the protocol by operating the device near the threshold for each data rate. From simulations, the computed upper bounds of EVM specification for QPSK, 16-QAM, and 64-QAM modulations are 35, 12, and 4.3%, respectively, for a BER compliance of $5e^{-4}$. While the data rate and modulations are changed by the higher-level protocol, the above described adaptive operation minimizes power consumption in the receiver by exploiting the EVM margins. The available margins for each modulation are a strong function of the individual circuit components of the receiver. Therefore, careful designs of tunable components in the receiver that exhibit power vs. performance trade-off maximize the power savings in a VIZOR receiver. The adaptive receiver shows significant power savings across different channel conditions with *a maximum savings going up to 4X (QPSK),* excluding the efficiency loss in the PMU.

9.4.1.5 VIZOR Transmitter

This subsection describes how the wireless transmitter can be adapted to the environment for low-power operation [14, 46].

Transmitter Adaptation

The PA is the most power hungry block in the transmitter. Hence VIZOR operation in the transmitter is targeted towards saving power in the PA. Figure 9.17 provides an illustration of the adaptive operation of the transmitter. In OFDM transmitter, there is a requirement of high P1dB specification (1 dB compression point) for the PA due to the high peak-to-average ratio (PAR) of the transmitted signal. This requirement of a high PAR ratio translates to high DC power consumption in the PA. To address this issue, a dynamic PAR reduction block (called companding) is used in the transmitter DSP that control the amount of PAR reduction adaptively under favorable channel conditions. This in turn reduces the P1dB requirements of the PA for the same output power level. Significant power savings can be obtained by co-tuning PAR reduction with adaptive PA operation depending upon the channel quality. Adaptive biasing of the PA is performed to save power (lower P1dB) by applying gate and drain bias though the power management blocks. The adaptive RF PA maintains its class of operation across different operating points, so that the linearity requirement and out of band spurious emission performance is maintained while the PA adapts for low-power operation.

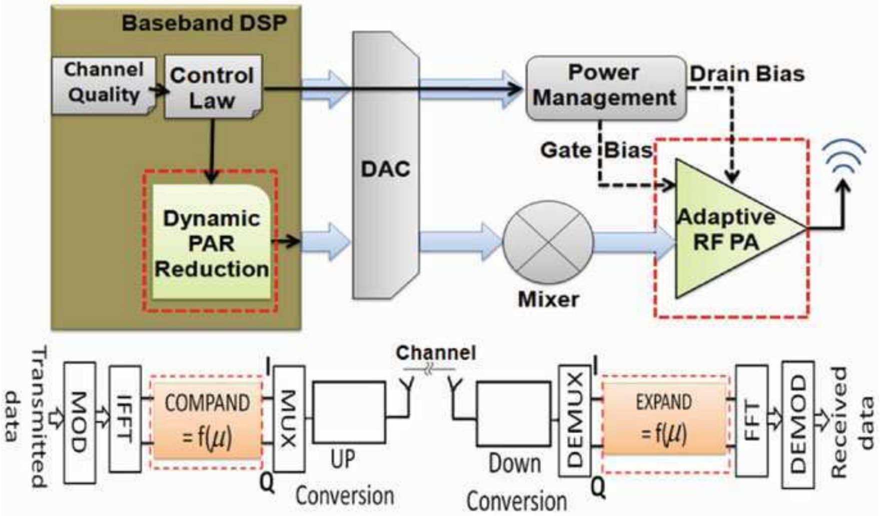


Fig. 9.17 Block diagram of the VIZOR transmitter [45]

Dynamic Companding and Adaptive PA Operation

Dynamic companding is done using the following formula,

$$x_{\mathrm{nc}} = K\frac{\mathrm{sign}(x) \times \ln\left[1 + \mu\left|\frac{x_n}{A}\right|\right]}{\ln[1 + \mu]}$$

$\mathbf{x_n}$ **= original signal**

$\mathbf{x_{nc}}$ **= companded signal** $\quad 0 \leq \left|\frac{x_n}{A}\right| \leq 1$

where K is the scaling factor and μ is the companding factor that is dynamically modified. The signal is retrieved by expanding in the receiver. The more the signal is companded, lesser the PAR, and hence the P1dB requirement of the PA reduces. On the other hand, as companding increases the signal becomes more and more susceptible to noise. The maximum amount of PAR reduction that can be achieved without compromising on BER for different channel conditions is obtained from system level simulations. For each channel the maximum achievable PAR reduction is found from the value of μ for which $\mathrm{EVM} = \mathrm{EVM_T}$. An adaptive PA that maintains its class of operation while adapting is used. The relaxed P1dB requirement is exploited by rebiasing the PA and saving power in real-time. The relation between the achievable PAR and the corresponding drain and gate voltages of the PA for minimum power operation is characterized prior hand, and used during online operation for adaptive tuning of the transmitter. During run time the real-time EVM information is fed back from BS to MS (refer Fig. 9.11) and is used for adapting the PA.

Power Savings

Figure 9.18 plots the P1dB and DC power consumption of the adaptive PA and comparable static PA across the different bias points. Under good channel conditions, a PAR reduction of 7.25 dB is possible while maintaining the required EVM for QPSK modulation. This translates to relaxation of P1dB requirement from 12.6 to 5.4 dBm for the adaptive PA. This lets the adaptive PA to operate at lesser DC power consumption. Up to 3*X* power savings can be achieved as seen in the figure. The gate and drain bias of the adaptive PA in terms of companding factor μ defines the locus of transmitter for environmental adaptation.

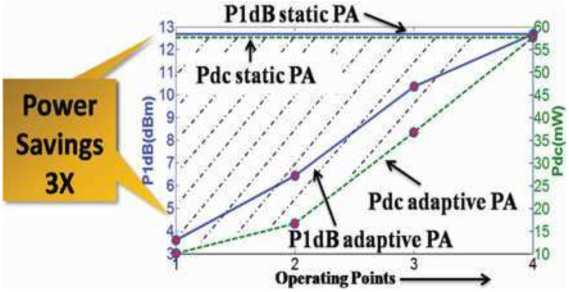


Fig. 9.18 Power consumption of static and adaptive PA and power savings [45]

9.4.1.6 Adaptive Low-Power VIZOR Operation Under Process Variation

Effect of Process Variation on Adaptive Low-Power Systems

With continued scaling the of device geometry in the nanometer regime the controllability of the fabrication process has reduced significantly, resulting in severe process variation. The effect of process variation in Analog/RF circuits shows up as failing of one or more specifications, resulting in classification of the device as faulty. This in turn reduces the yield of the system. To increase yield under process variation more built-in design margin is required for static designs, increasing power consumption. Hence low-power adaptive design methodologies become even more significant under sever process variation.

The previous subsections described the environment-adaptive VIZOR operations of the transmitter and receiver modules of MS for changing channel conditions. In both the cases, the device operation follows a pre-defined control law ("locus") that is obtained during the design/characterization phase. It should be noted that the optimal control law that is obtained is specific for a device and may not work well for another device in the manufacturing lot due to process variations. To address this issue, a test and process tuning (calibration) procedure must be performed during the production test phase to ensure that all the device operate at minimum power level while satisfying the system EVM/BER. The main components of such a *process variation tolerant environment-adaptive transceiver* are discussed below.

Process Sensing Using Test

Process sensing and tuning is performed using simple tests during the production testing phase of the product development cycle. Alternate testing methodology [47] provides simple and fast tests to estimate complex specification through simple test measurements using supervised learning techniques. Loopback-based alternate testing of RF transceivers has been shown as a low-cost production test technique where the performance-parameters (specifications) of a complex transceiver are estimated using a simple test approach. In loopback-based alternate testing the output of the transmitter is looped back to the receiver to estimate the receiver parameters. But loopback test suffers from accuracy issues because the looped back signal from the transmitter usually does not have high fidelity due to process variations in the transmitter itself. An adaptive testing technique such as shown in Fig. 9.19 enables process sensing using loopback test highly accurate.

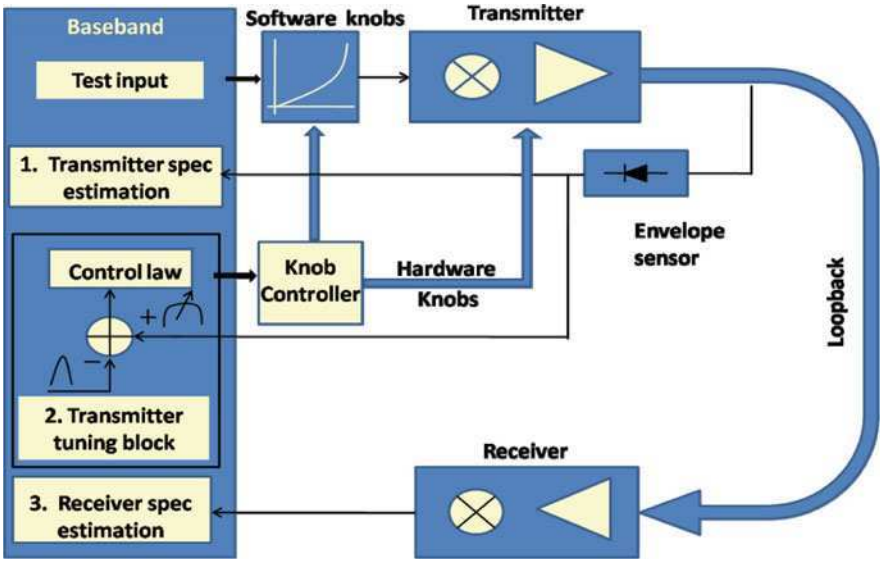


Fig. 9.19 ACT-based loopback testing approach [45]

Adaptive Calibration Test (ACT): the adaptive test methodology adopts a three-step procedure to ensure high accuracy of estimation.

- Test the transmitter for its specifications.
- Tune the transmitter for process variations by changing the available hardware and software knobs to ensure a near ideal operation of the transmitter
- Estimate the receiver specifications using loopback from the transmitter.

The tuning performed in this phase of product development cycle is to facilitate low-cost process test. Once the specifications are estimated accurately the tuning knobs can be reverted back to nominal operating conditions.

Tuning Technique: In ACT-based tuning technique an envelope detector is used to obtain information about the transmitter characteristics. A golden envelope response pertaining to an ideal transmitter instance is obtained before-hand.

Circuit Tuning: A simple gradient-based adaptive algorithm is used to dynamically control the circuit tuning parameters. The algorithm tunes the bias and supply values continually till the observed envelope of the system is within a certain tolerance limit in comparison to the golden envelope.

The distortion information obtained by comparing the observed envelope with the golden envelope is then used, to compute the coefficients of the correction polynomials, to correct the input of the system in such a way that the output of the system is in correspondence to near ideal operation (i.e., after tuning the observed envelope as close as possible to the golden envelope). Figure 9.20a shows the prediction of the transmitter gain of the DUT. Figure 9.20b shows the estimation of receiver gain without tuning the transmitter (conventional loopback approach). Figure 9.20c shows the receiver gain prediction for ACT-based loopback approach. Similar results for other specifications could be found in [48]. Using ACT both the transmitter and receiver specifications are predicted very accurately. But any other test/diagnosis technique would hold in this context. The process parameter sensed in steps (a) and (c) would now be used for performing process-tolerant VIZOR operation.

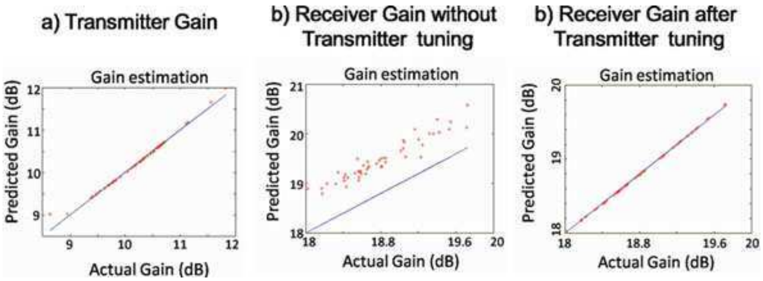


Fig. 9.20 Estimation plots for system gain

Low-Power Adaptation Under Process Variation

An illustration of process tuning procedure is given in Fig. 9.21. Process estimation using alternate test and tuning is performed during the production test phase. Figure 9.21 shows the process estimation and tuning flow was for a transceiver. The power optimal operation of a VIZOR transmitter or receiver depends on a pre-characterized locus as discussed before. Due to process variations in manufactured devices, the optimal locus is different for every device. Therefore, the purpose of process tuning is to identify the “right” locus for the DUT. The following steps

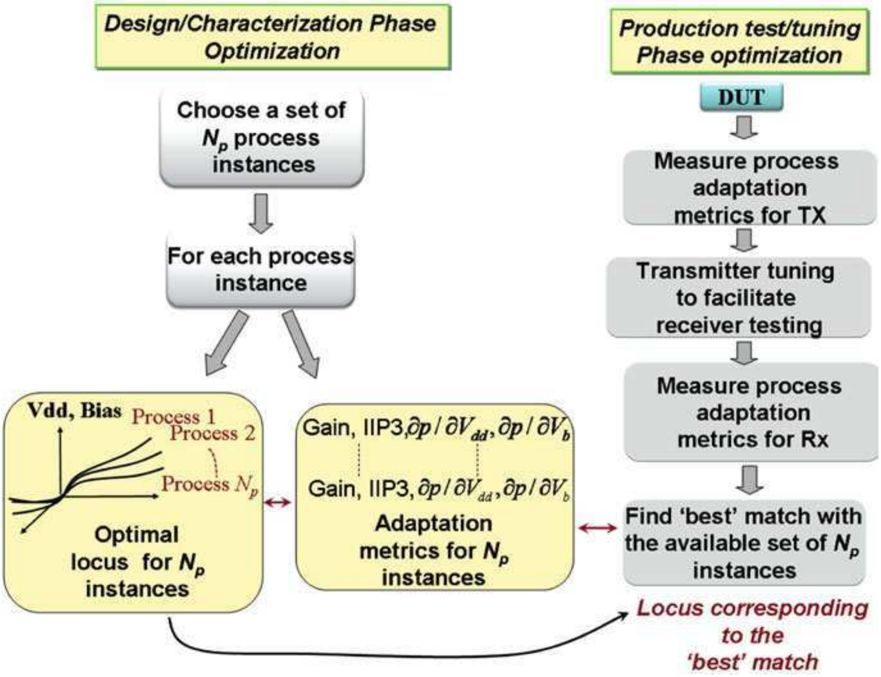

Design/Characterization Phase Optimization
Choose a set of N_p process instances
For each process instance
Vdd, Bias
Process 1
Process 2
Process N_p
Optimal locus for N_p instances
Gain, IIP3, $\partial p/\partial V_{dd}$, $\partial p/\partial V_b$
Gain, IIP3, $\partial p/\partial V_{dd}$, $\partial p/\partial V_b$
Adaptation metrics for N_p instances
Production test/tuning Phase optimization
DUT
Measure process adaptation metrics for TX
Transmitter tuning to facilitate receiver testing
Measure process adaptation metrics for Rx
Find 'best' match with the available set of N_p instances
Locus corresponding to the 'best' match

Fig. 9.21 Process sensing and estimation of optimal locus for minimum power operation of the DUT under process variation [45]

are performed to ensure this. A set of N_p process perturbed (nominal distribution) instances that adequately span the entire process space is generated during the design/characterization phase. For each of these instances, the optimal locus is obtained using the earlier described optimization. Key circuit-level specification values such as gain, IIP3 (third order intercept), phase noise, etc. and power sensitivities are measured and stored. Here, the power sensitivity refers to the power gradients w.r.t. changing tuning knob values – $\partial p/\partial V_{dd}$, $\partial p/\partial V_b$. The above-mentioned device characteristics, referred to as the *process-adaptation metrics* along with the optimal locus constitute completely to characterize a device. Moreover, the process-adaptation metrics of each device exhibit a strong correlation with the optimal locus for that device. If the process-adaptation metrics can be accurately measured for a device, the corresponding optimal locus can be estimated using a simple correlation-based mapping function. Here, a simple look-up table (LUT) based approach is used. A LUT consisting of the adaptation metrics along with optimal locus for the N_p instances is stored for future reference (product test/tuning phase).

During the production test/tuning phase, the process-adaptation metrics are first measured for the transmitter module in the DUT. Once the transmitter metrics are measured, the corresponding "best" fit locus is obtained by referring to the LUT and comparing the adaptation metrics of the DUT with the available set. Next, the transmitter is tuned to facilitate receiver testing as described earlier. Once this is done, the receiver adaptation metrics are estimated, which are in turn used to estimate the optimal locus for DUT. Thus, the “best” fit locus can be obtained for each device facilitating minimum power operation during run time.

Extra Power Savings Using Process Tuning

Figure 9.22a shows the DC power consumed by a process-perturbed instance before and after process tuning. Run-time operation of the device was first simulated. For this purpose, the device was adapted for changing channel conditions using a locus obtained for a nominal device. Later the device was tested and the optimal locus is obtained after estimation of process-adaptation metrics. The run-time operation was then simulated using the obtained optimal locus. As observed from Fig. 9.22a, when process was tuned, the device consumes lesser power for all the 12 different simulated channel conditions. The difference in device DC power consumption (Δp), while operating along the actual and estimated loci are calculated and plotted as an histogram for all the 50 instances. The Δps for most of the instances being close to zero shows the accuracy of process sensing and tuning approach.

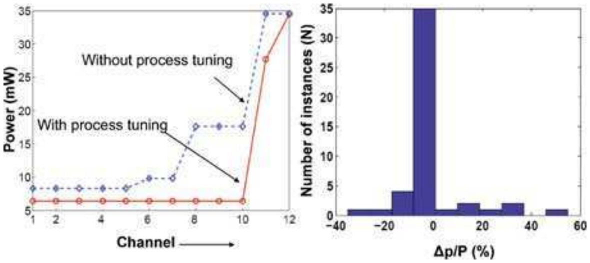


Fig. 9.22 (**a**) Power savings obtained with process tuning and (**b**) optimality of locus selection: histogram of Δp/P in % [45]

9.4.2 Power Conscious System Level Self-Healing Solution

The concept of system level healing is described with a case study of yield improvement for a mobile transmitter.

9.4.2.1 Effect of Process Variation and Built in Tunability

As described in previous sections the reduced controllability of the fabrication process in the nanometer regime reflects as significant variation of the important specifications from chip to chip. The yield of a system design starts to get affected severely as variation in the subsystems directly affects the system specifications. In this section we target to develop healing solutions for complete systems. The general methodology of *healing* a process skewed system is to have *built-in tunability* in the system and intelligently tune the system under variation such the system specifications come back within acceptable bounds [49–52].

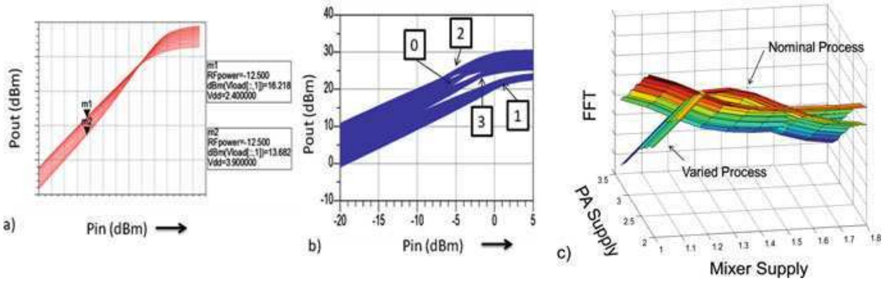


Fig. 9.23 (**a**) P_{out} vs. P_{in} for a PA over V_{dd} variation. (**b**) P_{out} vs. P_{in} over all combination of tuning knobs (supply and bias) for four process instances. (**c**) Gain of a transmitter over all tuning knob combinations for 2 process instances [52]

This calls for identifying proper tuning knobs in a design such that the tunable range obtained of the specifications concerned is close to or more than the variations of the same for a 3 sigma variation of the process parameters. This fact is illustrated for a PA in Fig. 9.23. Figure 9.23a shows the P_{out} vs. P_{in} for a PA with V_{dd} (supply voltage) varying within its tuning range. It can be seen that for higher supply voltages gain is lower (due to g_{m} being lower at high values of I_{d}), P1dB is higher and vice versa. So a gain vs. P1dB trade-off can be achieved by using supply as a tuning knob. Similarly even more tuning capability is achieved by using supply and bias of the PA as two tuning knobs simultaneously. Figure 9.23b plots the P_{out} vs. P_{in} for the PA over all combinations of supply and bias knob settings for 4 process instances, 0 being the nominal instance. It is to be noted that some of them (namely 2 and 3) exhibit "small parameter variation" [52] and can be tuned back completely by choosing the right tuning knob values. On the other hand there are instances (like 1) which exhibit "large parameter variation" and can only be tuned for its Gain and not P1dB. To understand how a specification changes with process and how it can be tuned back the Gain of an illustrative Transmitter (with tunable PA and Mixer) is shown in Fig. 9.23c for the nominal process and a varied process instance. For the nominal case the transmitter is to be biased at maximum gain. However, under variation this setting is not optimum for gain. So the job of the *healing algorithm* is to find out a new tuning knob setting such that the gain is closest to its nominal value for the varied process instance. Now there could be several such knob settings that provide the same gain. A *power conscious* self-healing system would choose the setting with minimum hit in power consumption. A *system level* power conscious self-healing framework would find a new setting that optimizes all the specifications for the system simultaneously while minimizing power.

9.4.2.2 System Level Healing Framework

The framework for power conscious system level self-healing is described below and is shown in Fig. 9.24. The methodology consists of the following key steps:

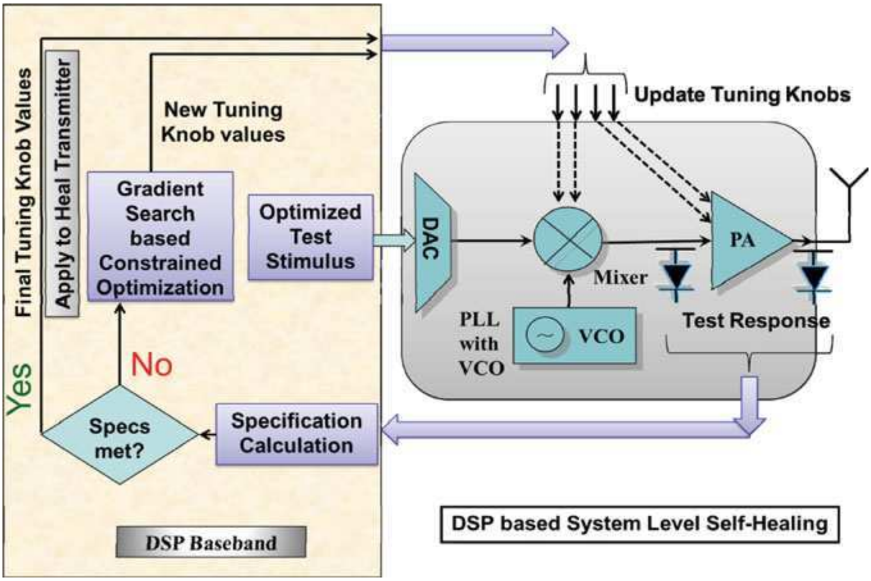


Fig. 9.24 System level power conscious self-healing framework

- The system is made up of tunable components with built in tuning knobs.
- During production test/characterization/tune phase any instance with non-nominal specifications are tuned to be as close as possible to the nominal one.
- An optimized test stimulus is applied to the system with nominal tuning knob settings. Important specifications are calculated from the test response using standard methods or alternate test methods [47] using built in sensors (as shown in Fig. 9.24) for faster response time.
- If the specifications are within the preset bounds the device passes and the current knob settings are to be applied while operation.
- If not, the device would have failed in static sense and should now be healed to try to pass it, so that the yield increases.
- A power conscious constraint optimization algorithm updates the knob settings and repeats this procedure until the best knob setting for the given process is achieved.
- The final setting found from the optimizations is applied to the system during runtime operation.
- At the end of the optimization a significant number of instances are expected to meet all of the specifications (as 2 and 3 in Fig. 9.23b); however, there would be some that would still not meet all the specifications (as 4 in Fig. 9.23b).

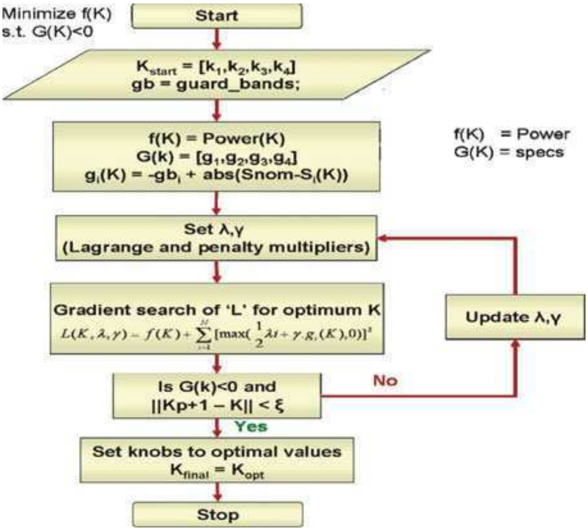
The detail of the constrained optimization used to perform the power conscious system level healing is described below.

9.4.2.3 Core Algorithm: Constrained Optimization

In this subsection an "Augmented Lagrange" based nonlinear optimization approach is described to search for the best possible way to turn a knob with the least impact on power. The details of the algorithm are shown in Fig. 9.25. A typical constrained optimization problem is formulated as follows:

$$\begin{aligned} &\text{Minimize :} \qquad f(x) \\ &\text{Subject to } G(x) \leq 0 \end{aligned} \tag{9.1}$$

Fig. 9.25 Power constrained healing algorithm [52]


Minimize f(K)
s.t. G(K)<0
Start
$K_{start} = [k_1,k_2,k_3,k_4]$
gb = guard_bands;
f(K) = Power(K)
$G(k) = [g_1,g_2,g_3,g_4]$
$g_i(K) = -gb_i + abs(Snom-S_i(K))$
f(K) = Power
G(K) = specs
Set λ,γ
(Lagrange and penalty multipliers)
Gradient search of 'L' for optimum K
$L(K,\lambda,\gamma) - f(K) + \sum_{i=1}^{N} [\max(\frac{1}{2}\lambda i + \gamma . g_i(K), 0)]^2$
Update λ,γ
Is G(k)<0 and
$||Kp+1 - K|| < \xi$
No
Yes
Set knobs to optimal values
$K_{final} = K_{opt}$
Stop

Where *f*(*x*) is the main objective function that needs to be minimized and the function set $G(x) = [g_1(x), g_2(x) \ldots g_n(x)]$ are the set of constraints that need to be satisfied while minimizing *f*(*x*). The key objective of this work is to improve the overall yield of the system with minimal impact on the power consumption. In this work the DC power consumption of the RF transmitter system across the supply and bias tuning knobs is formulated as the main objective function that needs to be minimized. The accompanying specs are formulated as multiple constraints.

Constraint Formulation: For a given process the nominal specification is taken to be *Snom* and the DUT's specification to be *S*. It is common to guard band the nominal specifications in order to trade-off multiple specs. Let *gb* be defined as the guard bands associated with the nominal specs. It is desirable that $abs(Snom - S) \leq ab$. The previous equation can be slightly modified as shown Equation (9.2) to represent a typical inequality constraint similar to Equation (9.1).

$$g_i(x) = -gb_i + \text{abs}(\text{S}_i\text{nom} - \text{S}_i) \leq 0 \tag{9.2}$$

The constraints considered are ***Gain,*** and distortion measures such as ***IIP2*** and ***IIP3***. A method frequently used to find solutions for problems like Equation (9.1) is the "Augmented LaGrangian method" that combines both the "penalty methods" and the "LaGrangian" methods for solving a constrained optimization problem. The augmented Lagrange function can be formulated in many different ways. Here, the function given by [53] is adapted to convert a constrained problem to an unconstrained function as shown in Equation (9.3).

$$L(x, \lambda, \gamma) = f(x) + \sum_{i=1}^{N} [\max(\frac{1}{2}\lambda i + \gamma . g_i(x), 0)]^2 \tag{9.3}$$

Where λi are the LaGrangian multipliers and the γ is the penalty parameter. A detailed discussion of this technique is beyond the scope of this paper. Detailed insights into the search technique can be found in [53]. In Equation (9.3) "x" refers to a unique combination of the tuning knob values. The update conditions for the λi variables can be derived to the expression shown below. Penalty values (γ) are increased if λ variables fail to update in a particular iteration.

A simple pseudo code of the algorithm is as follows:

(1) Start with an initial x_0, $\lambda = 0$, and $\gamma = 1$.
(2) Use gradient descent approaches [50] to solve for the minimum (x_k^*) of Equation (9.3).
(3) Update $\lambda_{k+1} = \max(\lambda_k + 2\gamma_k \cdot g(x_k^*), 0)$.
(4) Update $\gamma_{k+1} = 2 \cdot \gamma_k$ if λ has not changed from previous iteration.
(5) Iteratively optimize till $\left\| x_k^* - x_{k+1}^* \right\| < \xi$ or if a preset no of iterations are run.

Local Minima: While gradient search techniques do tend to get caught up in the local minimums, the penalty and the lambda functions ensure that the overall search approach recovers quickly from a local minimum. Also some advanced technique proposed in [54] could be used to avoid this problem for an extremely complicated surface.

Figure 9.26 shows the exhaustive specification surfaces for Gain, IIP2, and IIP3. Figure 9.26 also shows the cost function surface (which is nothing but Equation (9.3) evaluated across all the knobs values "x") across the tuning knobs for the penultimate iteration particular value of λ and γ before convergence. It can be seen from the figures that the cost function has evolved to accommodate distinct minimum located at a supply voltage of 1.8 and bias of 0.6 that satisfies all the constraints with the least impact on the power consumption.

9.4.2.4 Yield Improvement: A Transmitter Case Study

In this section the tuning results for the augmented LaGrange based is presented.

The yield analysis of the lot before and after *self-healing* is shown above. Gain, IIP2, and IIP3 were used for the yield calculation. The bounds for gain, IIP2, and IIP3 were fixed as shown in Fig. 9.27. It shows the yield plots before and after tuning for the transmitter and the receiver. Based on the given bounds the yield

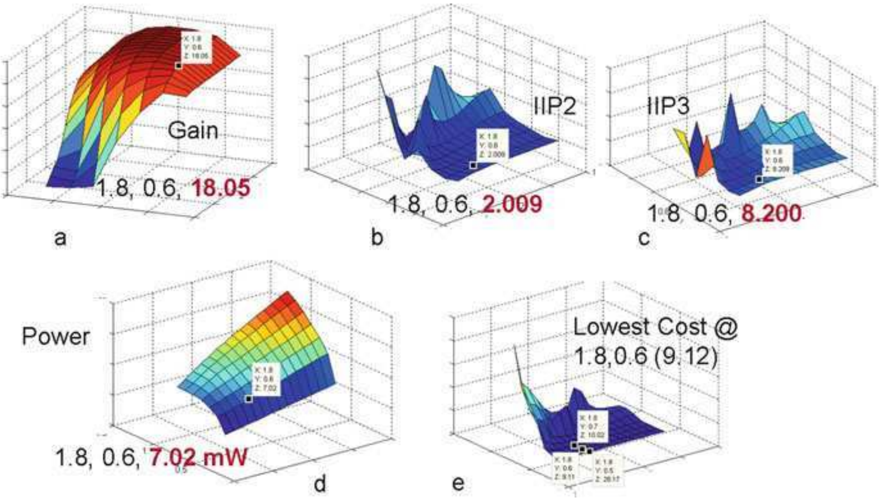


Fig. 9.26 Surface plots for specs and Aug-Lagrange cost [52]

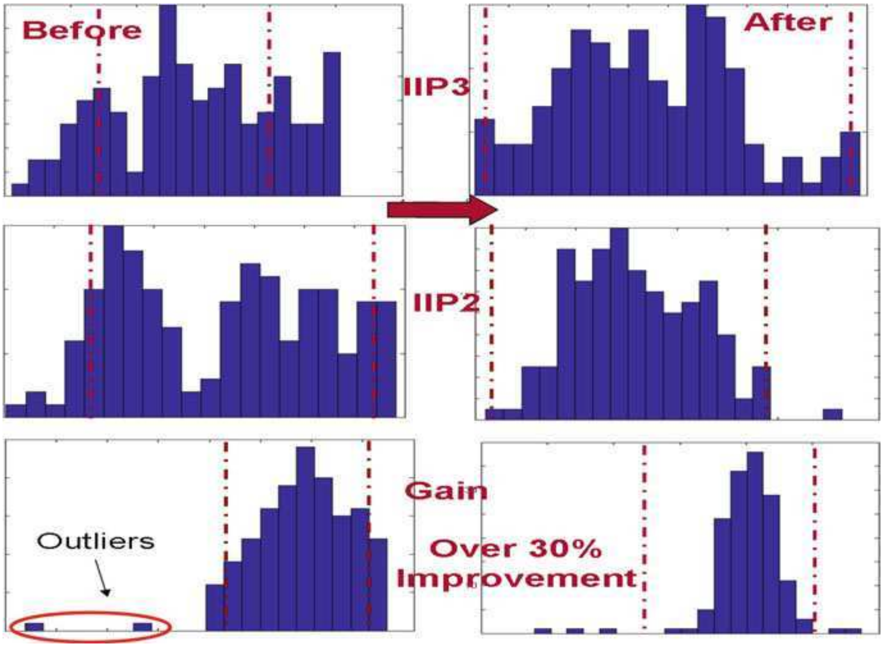


Fig. 9.27 Yield improvement results for tuning of transmitter using augmented LaGrange approach [52]

was observed to be 61.59% before tuning and 96.02% after tuning for the transmitter process lot thereby resulting in a yield improvement of 34.43%, showing the effectiveness of the power conscious self-healing approach.

9.5 Conclusion

This chapter discusses the problems in Analog/RF systems in the era of high integration and severe process variation. Power consumption and variability tend to dominate design choices in this era. This chapter provides an overview of low power and process tolerance techniques in Mixed Signal/RF circuits both in component level as well as system level. Several possible tuning knobs have been identified in important MS/RF components. Component level tuning for process as well as low power has been described. System level solutions that use the component level tuning knobs to provide increased controllability are discussed. As an example, a system level solution is discussed for adaptive low power under dynamically changing environment and process-tolerant solution for such an adaptive system was provided. Finally the chapter ends with a system level self-healing solution for process tolerance of complete wireless systems.

Acknowledgment The authors would like to thank Rajarajan Senguttuvan, Donghoon Han, and Shyam Devarakond for their valuable contributions to the content presented in this chapter. The authors would also like to thank Hasnain Lakdalwa and Krisnamurthy Soumyanath from Intel, Oregon for allowing the reuse of graphics from their work.

References

1. http://cseweb.ucsd.edu/classes/wi10/cse241a/slides/Ch1_Introduction.pptx
2. Borkar S, Karnik T, De V (2004) Design and reliability challenges in nanometer technologies, DAC ’04, p. 75
3. The international technology roadmap for semiconductors ITRS WEBSITE. [Online]. Available: http://public.itrs.net/Links/2009ITRS/2009Chapters_2009Tables/2009_Yield.pdf (accessed 4/12/2010)
4. Mukhopadhyay S, Kim K, Mahmoodi H, Roy K (Jun 2007) Design of a process variation tolerant self-repairing SRAM for yield enhancement in nanoscaled CMOS. IEEE J Solid-State Circuits 42(6):1370–1382
5. Azizi N, Khellah MM, De V, Najm FN (2005) Variations-aware low-power design with voltage scaling. DAC’ 05, pp 529–534
6. Chen T, Naffziger S (Oct 2003) Comparison of adaptive body bias (ABB) and adaptive supply voltage (ASV) for improving delay and leakage under the presence of process variation. IEEE Trans VLSI Syst 11(5):888–899
7. Kim CH, Roy K, Hsu S, Krishnamurthy R, Borkar S (Jun 2006) A process variation compensating technique with an on-die leakage current sensor for nanometer scale dynamic circuits. IEEE Trans VLSI Syst 14(6):646–649
8. Kimand T, Kim B (Apr 2006) Post-linearization of cascade CMOS low noise amplifier using folded PMOS IMD sinker. IEEE Microw Wirel Compon Lett 16(4):182–184
9. Sivonen P, Vilander A, Parssinen A (Sep 2004) A gain stabilization technique for tuned RF low-noise amplifiers. IEEE Trans Circuits Syst I: Reg Papers 51(9):1702–1707
10. Chen F, Weber RJ (2004) A novel process-variation insensitive network for on-chip impedance matching. In: IEEE international symposium on communications and information technology, 2004 (ISCIT 2004), pp 43–46
11. Tee YT et al (2003) Design techniques to combat process, temperature and supply variations in Bluetooth RFIC. In: IEEE radio frequency integrated circuits (RFIC 2003) Symposium, pp 551–554

12. Han D, Kim BS, Chatterjee A (Feb 2010) DSP-driven self-tuning of RF circuits for process-induced performance variability. IEEE Trans VLSI Syst 18(2):305–314
13. Sen S, Chatterjee A (2008) Design of process variation tolerant radio frequency low noise amplifier. In: IEEE international symposium on circuits and systems, 2008 (ISCAS 2008) pp 392–395, 18–21 May 2008
14. Sen S, Senguttuvan R, Chatterjee A (Jan 2008) Concurrent PAR and power amplifier adaptation for power efficient operation of WiMAX OFDM transmitters. In: IEEE radio and wireless symposium, pp, 21–24
15. Das T, Gopalan A, Washburn C, Mukund PR (Dec 2005) Self-calibration of input-match in RF front-end circuitry. IEEE Trans Circuits Syst II Exp Briefs 52(12):821–825
16. Palaskas Y, Taylor SS, Pellerano S, Rippke I, Bishop R, Ravi A, Lakdawala H, Soumyanath K, (Aug 2006) A 5-GHz 20-dBm power amplifier with digitally assisted AM-PM correction in a 90-nm CMOS process. IEEE J Solid-State Circuits 41(8):1757–1763
17. Malla P, Lakdawala H, Kornegay K, Soumyanath K, (2008) A 28 mW Spectrum-sensing reconfigurable 20 MHz 72dB-SNR 70dB-SNDR DT ΔΣ ADC for 802.11n/WiMAX Receivers. In: IEEE international solid-State Circuits Conference, 2008 (ISSCC 2008). Digest of Technical Papers, pp 496–631, 3–7 Feb 2008
18. Senguttuvan R, Sen S, Chatterjee A (2008) Concurrent multi-dimensional adaptation for low-power operation in wireless devices. In: 21st international conference on VLSI design, 2008 (VLSID 2008) pp 65–70, 4–8 Jan 2008
19. (2004) IEEE standard for local and metropolitan area networks part 16: air interface for fixed broadband wireless access systems, pp 1–857. Available: http://standards.ieee.org/getieee802/download/802.16-2004.pdf (accessed 9/15/2010)
20. Gil-Garcia A. Output power-control loop design for GSM mobile phones. Agilent technologies. Available: http://www.analogzone.com/hft_1206.pdf (accessed 3/10/2010)
21. Agilent ACPM-7891Tri-Band power amplifier module EGSM, DCS and PCS multi-slot GPRS. Data Sheet and application note. Available: http://www.datasheetcatalog.org/datasheet2/9/0o0o1a6ksi81keyca6y2josge5py.pdf (accessed 3/10/2010)
22. A multi-band GSM/GPRS power amplifier controller. Application Brief 122. Available: http://www.national.com/appbriefs/files/AppBrief122.pdf (accessed 3/10/2010)
23. Control loop design for GSM mobile phone applications. White paper, avago technologies. Available:http://avagotech.com/docs/AV02-2414EN (accessed 3/10/2010)
24. GSM power control and power class. *Tutorial* Available: http://www.radio-lectronics.com/info/cellulartelecomms/gsm_technical/power-ontrol-classes-amplifier.php
25. Minnis BJ, Moore PA, Whatmough PN, Blanken PG, van der Heijden MP (Jan 2009) System-efficiency analysis of power amplifier supply-tracking regimes in mobile transmitters. IEEE Trans Circuits Syst I Regular Pap 56(1):268–279
26. Pan H-I, Rincon-Mora GA (Jun 2006) Asynchronous nonlinear power-tracking supply for power efficient linear RF Pas. Int Conf Commun Circuits Syst Proc 4:2531–2535, 25–28
27. Hanington G, Pin-Fan Chen, Asbeck PM , Larson LE (Aug 1999) High-efficiency power amplifier using dynamic power-supply voltage for CDMA applications. IEEE Trans Microw Theory Tech 47(8):1471–1476
28. Wang F, Kimball DF, Lie DY, Asbeck PM, Larson LE (Jun 2007) A monolithic high-efficiency 2.4-GHz, 20-dBm SiGe BiCMOS envelope-tracking OFDM Power Amplifier. IEEE J Solid-State Circuits 42(6):1271–1281
29. Wang F, Yang AH, Kimball DF, Larson LE, Asbeck PM (Apr 2005) Design of wide-bandwidth envelope-tracking power amplifiers for OFDM applications. IEEE Trans Microw Theory Tech 53(4) Part 1, Page(s):1244–1255
30. Presti CD, Carrara F, Scuderi A, Asbeck PM, Palmisano G (Jul 2009) A 25 dBm digitally modulated CMOS power amplifier for WCDMA/EDGE/OFDM with adaptive digital predistortion and efficient power control. IEEE J Solid-State Circuits 44(7): 1883–1896

31. Singhal N, Pamarti S (Apr 2010) A digital envelope combiner for switching power amplifier linearization. IEEE Trans Circuits Syst II Exp Briefs 57(4):270–274
32. Sahu BRincón-Mora GA (Jan 2004) A high-efficiency linear RF power amplifier with a power-tracking dynamically adaptive buck-boost supply. IEEE Trans Microw Theory Tech 52(1): 112–120.
33. Ernst D, Das S, Lee S, Blaauw D, Austin T, Mudge T, Kim NS, Flautner K (Nov-Dec 2004) RAZOR: circuit-level correction of timing errors for low-power operation. IEEE Microw 24(6):10–20
34. Burd TD, Pering TA, Stratakos AJ, Brodersen RW (Nov 2000) A dynamic voltage scaled microprocessor system. IEEE J Solid-State Circuits 35(11):1571–1580
35. Abidi A, Pottie GJ, Kaiser WJ (Oct 2000) Power-conscious design of wireless circuits and systems. Proc IEEE 88(10):1528–1545
36. Perumana BG, Chakraborty S, Lee CH, Laskar J (Jun 2005) A fully monolithic 260-_W, 1-GHz subthreshold low noise amplifier. IEEE Microw Wirel Compon Lett 15(6):428–430
37. Tasic A, Lim S-T, Serdjin WA, Long JR (Feb 2007) Design of adaptive multi-mode RF front-end circuits. IEEE J Solid State Circuits 42(2) 313–322
38. Tasic A. Serdjin WA, Long JR (2006) Adaptive multi-standard circuits and systems for wireless communications. IEEE Circuits Syst Mag 6(1): pp 29–37
39. Brodersen B, Davis WR, Yee D, Zhang N (2002) Wireless systems-on-a-chip design. In: Proceedings of international symposium on quality electronic design, 18–21 March p. 221
40. Woesner H, Ebert JP, Schlager M, Wolisz A (1998) Power saving mechanisms in emerging standards for wireless LANs: the MAC layer perspective. IEEE Personal Commun Syst 5(3):40–48
41. Debaillie B, Bougard B, Lenoir, G, Vandersteen G, Catthoor F (2006) Energy-scalable OFDM transmitter design and control. In: 43rd IEEE design automation conference, July 24–28, pp 536–541
42. Sen S, Senguttuvan R, Chatterjee A (2007) Feedback driven adaptive power management for minimum power operation of wireless receivers. In: 14th IEEE international conference on electronics circuits and systems, 2007 (ICECS 2007), pp 1019–1022, 11–14 Dec 2007
43. Senguttuvan R, Sen S, Chatterjee A (2007) VIZOR: virtually zero-margin adaptive RF for ultra-low-power wireless communication. IEEE ICCD, Lake Tahoe, USA
44. Senguttuvan R, Sen S, Chatterjee A (Sept 2008) Multidimensional adaptive power management for low-power operation of wireless devices. IEEE Trans Circuits Syst II Exp Briefs 55(9):867–871
45. Sen S, Natarajan V, Senguttuvan R, Chatterjee A (2008) Pro-VIZOR: process tunable virtually zero margin low power adaptive RF for wireless systems. In: 45th ACM/IEEE Design automation conference, 2008 (DAC 2008), pp 492–497, 8–13 June 2008
46. Sen S, Senguttuvan R, Chatterjee A Channel-adaptive concurrent companding and bias control for efficient power amplifier operation. In: IEEE transaction on circuits and systems I (under review)
47. Variyam PN, Cherubal S, Chatterjee A (March 2002) Prediction of analog performance parameters using fast transient testing. IEEE Trans on Comput Aided Des Integr Circ Syst 21(3):349–361
48. Natarajan V, Senguttuvan R, Sen S, Chatterjee A (2008) ACT: adaptive calibration test for performance enhancement and increased testability of wireless RF front-ends. In: 26th IEEE VLSI test symposium (VTS 2008), pp 215–220, April 27 2008–May 1 2008
49. Chatterjee A, Han D, Natarajan V, Devarakond S, Sen S, Choi H, Senguttuvan R, Bhattacharya S, Goyal A, Lee D, Swaminathan M (2009). Iterative built-in testing and tuning of mixed-signal/RF systems. In: IEEE International conference on computer design, 2009 (ICCD 2009), pp 319–326, 4–7 Oct 2009
50. Natarajan V, Devarakond SK, Sen S, Chatterjee A (2009) BIST driven power conscious post-manufacture tuning of wireless transceiver systems using hardware-iterated gradient search. Asian test symposium, 2009 (ATS '09), pp 243–248, 23–26 Nov 2009

51. Devarakond SK, Natarajan V, Sen S, Chatterjee A (Jun 2009) BIST-assisted power aware self-healing RF circuits. In: IEEE 15th international mixed-signals, sensors, and systems test workshop, 2009 (IMS3TW '09), pp 1–4, 10–12 Jun 2009
52. Natarajan V, Sen S, Devarakond SK, Chatterjee A (2010) A holistic approach to accurate tuning of RF systems for large and small multi-parameter perturbations. In: IEEE VLSI test symposium, 2010 (VTS '10) pp 331–336
53. Snyman JA (2005) Practical mathematical optimization. Springer. New York, NY
54. Shang Y, Wah BW (1998) A discrete Lagrangian Based Global Search Method for Solving Satisfiability Problems. J Global Optim 12(1):61–99. DOI: 10.1023/A: 1008287028851

Part IV
Low-Power and Robust Reconfigurable Computing

Chapter 10
Low-Power Techniques for FPGAs

Nikil Mehta and André DeHon

Abstract Field-programmable gate arrays (FPGAs) are reconfigurable devices that can be programmed after fabrication to implement any digital logic. As such, they are flexible, easy to modify in-field, and cheaper to use than manufacturing a customized application-specific integrated circuit (ASIC). However, this programmability comes at a cost in terms of area, performance, and perhaps most importantly power. As currently manufactured, FPGAs are significantly less power efficient than ASICs. Fortunately, in the last decade concentrated attention to power consumption has identified many approaches to power reduction. This chapter surveys the techniques and progress made to improve FPGA power efficiency.

10.1 Introduction

As integrated circuit design becomes more complex, time-consuming, and expensive, FPGAs offer several advantages over ASICs: faster time to market, lower non-recurring engineering costs, early access to advanced technology, and post-fabrication programmability that allows task specialization and field upgrades. However, the flexibility and programmability of FPGAs come at a cost in terms of area, performance, and power. We will see in Section 10.2.2 that the power dissipation gap between FPGAs and ASICs has been estimated to be 7–14× on average [30]. This gap becomes increasingly important as energy, power, and power density become dominating concerns for electronic systems. Understanding and reducing this near order of magnitude gap has been the subject of intense research by the FPGA community.

N. Mehta (✉)
Department of Computer Science, California Institute of Technology, Pasadena, CA, USA
e-mail: nikil@caltech.edu

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_10,

While many of the low-power design techniques developed for ASICs mentioned in prior chapters can be directly applied to FPGAs, there are a number of FPGA centric optimizations that yield additional power savings. These techniques target specific characteristics of FPGAs across all levels of design. By combining and exploiting these techniques we can narrow the power gap.

This chapter will outline the challenges in decreasing FPGA power dissipation (Section 10.1) and progress made in FPGA power reduction across all levels of design, from top to bottom: CAD (Section 10.3), architecture (Section 10.4), circuits (Section 10.5), and devices (Section 10.6). Section 10.7 summarizes the key developments and future challenges in FPGA power reduction.

10.2 FPGAs Power Dissipation

In the next two sections we provide a brief overview of FPGA architecture, sources of power dissipation (both qualitatively and quantitatively), and the power gap between FPGAs and ASICs.

10.2.1 FPGA Overview

A conventional, island-style FPGA can be viewed as an array of configurable logic blocks (CLBs) connected by programmable interconnect (Fig. 10.1). Desired logic functions are programmed into look up tables (LUTs) (Fig. 10.2a) inside the CLBs and wired together by programming switches (Fig. 10.2b) in connection boxes and switch boxes in the interconnect. Any circuit can be realized; however, to support this programmability the FPGA must provision significantly more resources than those required by a custom designed and fabricated implementation. This resource overhead costs area, performance, and power.

As with ASICs, power dissipation in an FPGA can be separated into static and dynamic power (assuming short circuit power is negligible); we can express power

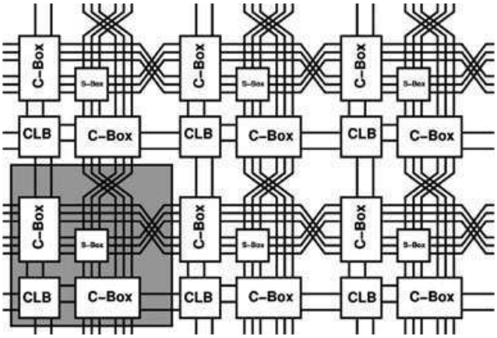


Fig. 10.1 Island-style FPGA architecture

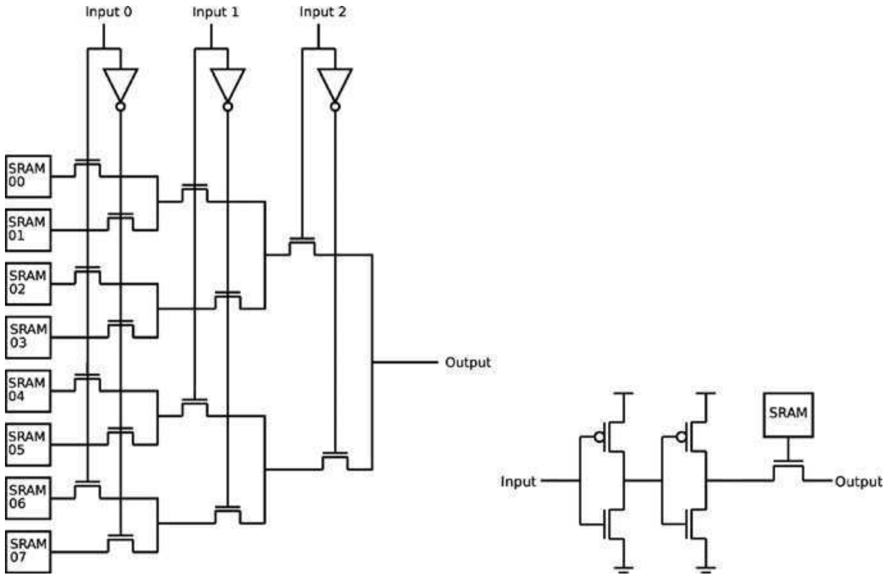


Fig. 10.2 FPGA logic and switch circuits. (**a**) 3-input LUT (**b**) Switch

as the sum of the canonical power equations (Chapter 2, Equations (2.2) and (2.10)) over all N transistors in the chip:

$$P_{\text{dynamic, chip}} = (V_{\text{dd}})^2 f \left(\sum_{i=0}^{N} \alpha_i C_i \right) \tag{10.1}$$

$$P_{\text{static, chip}} = V_{\text{dd}} \left(\sum_{i=0}^{N} I_{\text{leak},i} \right) \tag{10.2}$$

where α is switching probability, C is switched capacitance, f is frequency, I_{leak} is average leakage current, and V_{dd} is the power supply. An important element missing from this model is the amount of power dissipated due to spurious logic transitions, or glitches. The probability α_i here assumes that node i is switching at most once per cycle, and that every switching event is necessary to perform the desired logic function. In reality, because combinational paths are often unbalanced, internal nodes in a combinational block may switch several times per cycle, depending on the inputs to the block. We will revisit glitch power in Section 10.3.6.

The main sources of power dissipation in an FPGA are the logic, interconnect, clock network, and configuration memory. Modern FPGAs also consume non-negligible power in embedded blocks (memories, multipliers, etc.) and in I/O drivers. Several studies have estimated the power dissipated in each of these elements to provide a breakdown of both static and dynamic power dissipation [31, 48, 52]. Each study arrives at similar breakdowns for power dissipation; Fig. 10.3 shows

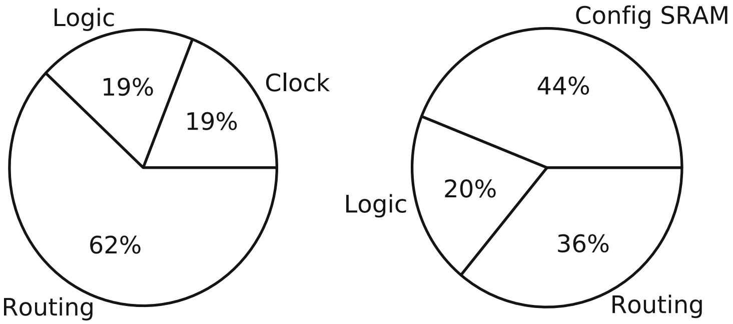


Fig. 10.3 Measured power breakdown for 90-nm Xilinx Spartan-3 [52]. (**a**) Dynamic Power (**b**) Static Power

the most recently published results for a 90-nm Xilinx Spartan-3 [52]. Dynamic and static power charts are shown; in this particular device dynamic power is typically an order of magnitude larger than static power. However, for more recent, high performance devices static power is expected to dominate total power. In Fig. 10.3a we see that interconnect dominates dynamic power, contributing 62%. Clock and logic power are the next two largest contributers and are roughly equivalent at 19%. For static power (Fig. 10.3b) configuration bits contribute the largest amount of leakage at 44%, as every switch requires a configuration bit and every k-input LUT requires 2^k bits. Routing and logic transistors also leak significantly. If we assign the static power of configuration bits to their sources (either interconnect or logic), interconnect static power will dominate. Hence, interconnect is the primary source of both dynamic and static power dissipation.

While interconnect can often be a major driver of power in ASICs, the need for active elements to buffer and switch interconnect in FPGAs makes it much clearer how much energy goes into the interconnect. This effect is exacerbated by the need to provision adequate programmable interconnect resources to support a wide range of designs. In the next section we will examine the cost of this programmability in how FPGA power compares to ASIC power.

10.2.2 FPGA/ASIC Power Gap

While the configurability, time-to-market, and low NRE advantages of FPGAs are compelling, many power-sensitive designs are still drawn to ASICs. When solving identical problems, ASICs consume less power.

The main source of power dissipation in both dynamic and static cases is the added capacitance associated with programmability. This additional capacitance comes both from used and potential switches in the interconnect, and from longer wires that result from the area overhead for programmable switches and configuration memories. The area required to implement an FPGA LUT and flip-flop using standard cells will be similar to the FPGA's LUT and flip-flop area [30]. However, from Fig. 10.1 we see that to connect two logic elements, instead of using and driving a single wire as in the ASIC case, in the FPGA we may need to route signals through several interconnect switches, each of which drives the input capacitance of several other switches.

For the static power case, if we assume that all transistors have identical characteristics (i.e. no variation), then I_{leak} is the same for all devices (Chapter 2, Equation (2.10)), and consequently P_{static} (Equation (10.2)) depends solely on N, or area. The main source of area overhead in an FPGA is again the programmable interconnect switches along with their configuration bits, which are not needed in an ASIC.

Kuon and Rose [30] begin to quantify the FPGA/ASIC power gap by synthesizing a set of benchmark circuits for both a 90-nm FPGA and a 90-nm ASIC process using a standard, commercial EDA flow without any specific power optimizations. They compared area, delay, and dynamic power of the ASIC to that of the FPGA in several different configurations: logic only and logic with different combinations of modern embedded structures (memory and DSPs). Table 10.1 summarizes their results, showing that on average a logic only FPGA without any optimizations utilizes 14× the dynamic power of a process equivalent ASIC. With non-configurable embedded memories and DSPs this gap decreases to 7×. Embedded elements do not contain programmable interconnect, saving capacitance and hence dynamic power. When comparing static power, they found that the gap is roughly correlated (correlation coefficient of 0.8) to the area overhead of $18 - 32\times$, with embedded blocks again reducing power overhead.

Table 10.1 FPGA/ASIC gap [30]

	Ratio (FPGA/ASIC)			
Metric	Logic only	Logic & DSP	Logic & memory	Logic, memory & DSP
Area	35	25	33	18
Delay	3.4	3.5	3.5	3.0
Dynamic power	14	12	14	7.1

The Kuon and Rose study carefully guarantees that the FPGA and ASIC are implemented in the same technology and solving the same problem. This provides a useful baseline for comparison. FPGAs, however, can typically exploit two advantages that are not expressed in this baseline. First, FPGAs may be able to use a smaller feature size technology that has lower capacitance. For example, state-of-the-art FPGAs are now manufactured in 40-nm technologies, while many ASICs are still using 130-nm technology because of the cost and difficulty of deep submicron design. Second, an FPGA may be able to solve a more specialized problem than the ASIC. For some problems it is possible to configure the FPGA and solve a specific instance of a problem rather than all instances. For example, ASIC filters must generally support programmable coefficients, while the FPGA implementation can be specialized to the particular coefficients needed for a special programming task. This results in substantial area reduction for the FPGA with commensurate power reduction [14] [23, chapter 22]. A 30% power advantage due to a smaller feature size technology coupled with an 80% power reduction due to specialization to a specific instance of the problem reduces FPGA power to $(1 - 0.3) \times (1 - 0.8) = 0.14$ of the same-problem and same-technology power and could close a 7× raw power gap.

While FPGAs consume more power than equivalent ASICs for identical technologies and solution implementations, FPGAs can consume significantly less power than conventional CPUs when performing identical tasks in identical technologies. CPUs put substantial energy into instruction switching, cycling of large memories, and control circuitry aimed at latency reduction. Their wordwide architecture also prevents them from matching their computation to the particular needs of a problem. For example, Abnous et al. compare energy consumption on a low-power embedded ARM processor to an FPGA for digital-signal processing functions, showing that the processor consumes over 15× the energy of the FPGA [2]. A generalized comparison between the FPGA/CPU as in the Kuon FPGA/ASIC study is still future work.

10.3 CAD

FPGA CAD optimizations have the potential to reduce costs without demanding any changes in the FPGA itself. In that sense, any gains they offer are immediately applicable without redesigning the FPGA or designing and fabricating a new family of chips. Since they do not change the physical chip, any tradeoffs arising at the mapping phase are elective, allowing the consumer to select the tradeoff that is appropriate to the problem. Consequently, there is a wealth of recent researches aimed at optimizing FPGA CAD algorithms to synthesize and map circuits that dissipate less power. The next several sections summarize these efforts in the FPGA CAD from start to finish: technology mapping, clustering, place, and route. Most of these techniques focus on reducing dynamic energy; however, one (Section 10.3.3) reduces static power.

Standard, technology-independent CAD transformations are used for both ASICs and FPGAs to reduce area, delay, and energy. While the FPGA and ASIC algorithms differ mainly in their relative cost models, in some cases unique FPGA features create both additional challenges and opportunities for optimization. Originally, many existing FPGA CAD mapping algorithms were designed to optimize area and delay, not to explicitly reduce energy consumption. Fortunately, many of these algorithms are already based around a cost function, and this function can be adapted to target energy by adding terms accounting for the energy impact of choices and transformations. A common theme in many of these optimizations is to exploit implementation freedom to reduce the capacitance associated with high activity (α) nodes.

10.3.1 Technology Mapping

FPGA designs typically begin with a netlist of gates and registers created by hand or more commonly compiled from a high-level language (e.g., VHDL, Verilog).

The technology mapping step of the FPGA CAD flow transforms this netlist into an FPGA targeted netlist of k-input LUTs and flip-flops. Algorithms partition the gate netlist into pieces and determine how to best "cover" sets of gates with LUTs. Technology mappers are most often concerned with minimizing area by reducing the total number of mapped LUTs and with minimizing delay by reducing logic depth (i.e., the maximum number of LUTs in series).

The goal of low-power technology mapping is to reduce dynamic power (Equation (10.1)) by either minimizing the switching activity of connections between LUTs (α_i) or reducing the total number of these connections and therefore the overall capacitance (C_i). Because connections between LUTs are highly capacitive due to the large number of switches attached to a wire segment (Fig. 10.1), it would be beneficial to both activate these segments as infrequently as possible and to reduce the total number of segments.

Previous work has reduced interconnect switching activity by identifying the highest activity connections between gates and mapping in such a way that these nodes are internal to a LUT [17, 33, 40]. The ability to hide internal nodes in compute blocks is an unique opportunity that arises in the FPGA that is not present in ASIC optimization. Figure 10.4 demonstrates an example of two ways of performing technology mapping on a set of gates with nets annotated with switching activity factors. Both mappings produce the same number of LUTs (area) and the same logic depth (delay). However, the power optimized mapping hides the $\alpha = 0.90$ node within a LUT, reducing the mapped circuit switching activity and power.

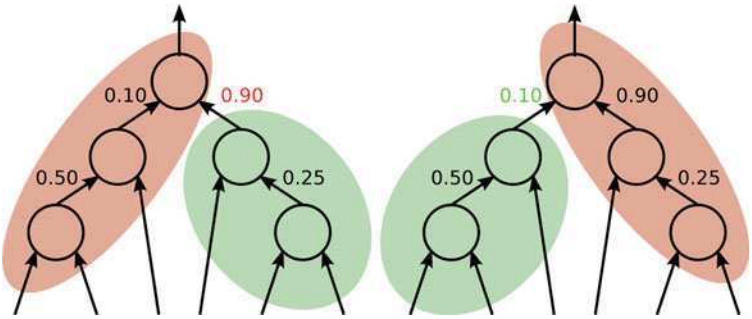


Fig. 10.4 Low-power FPGA technology mapping. (**a**) Unoptimized for power; (**b**) Optimized for power

Other work has reduced the total number of interconnect segments by limiting duplication [4], a common technique used in technology mapping. Figure 10.5 demonstrates an example of mapping a set of gates to 4-LUTs with and without duplication. Gates X and Y must be covered with two separate LUTs because LUTs only have one output. Gate B fans out to both X and Y, and without duplication must be covered with a third LUT. However, if we duplicate B we can place one B gate in the X LUT and the other in the Y LUT, reducing the total number of LUTs by one. Duplication is necessary for depth optimal mapping and can further aid in hiding high activity nodes within LUTs. Unfortunately, duplication can sometimes result

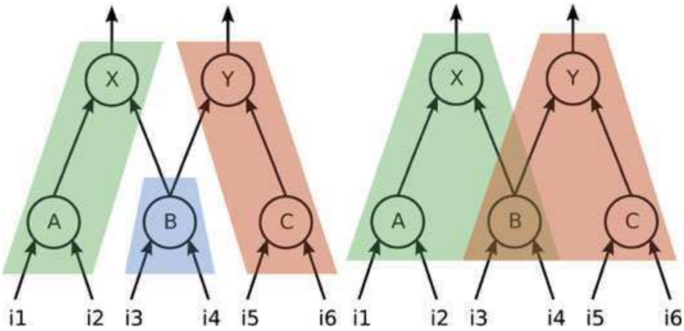


Fig. 10.5 Duplication in technology mapping [4]

in a net energy increase because of increased routing capacitance. In our example, without duplication signals i1–i6 must only connect to six LUT input pins. With duplication, however, inputs i3 and i4 must connect to two additional input pins, for a total of 8 pins. Since these signals are arriving from highly capacitive general purpose interconnect, dynamic energy may increase. This can be avoided by only performing duplication when it results in a net power reduction.

In general, low-power technology mapping has been shown to yield approximately $7.6-17\%$ power savings using both switching activity reduction and limited duplication [33, 40].

10.3.2 Clustering

After producing a netlist of LUTs and flip-flops from technology mapping, these elements must be clustered together to produce a netlist of CLBs. Modern FPGAs typically have between 4 and 10 LUTs per CLB, with each LUT having the option to register its output. Connections between LUTs are highly capacitive when those LUTs are located in different CLBs as signals must travel through global interconnect switches (Fig. 10.1). However, when LUTs are located within the same CLB, connections use local interconnect that has lower capacitance, which is more power efficient. Figure 10.6 shows an example of how inter-CLB connections are organized.

Conventional clustering algorithms attempt to either minimize area by reducing the total number of CLBs by packing them as full as possible, or to minimize delay by packing LUTs on the critical path within shared clusters. A commonly used clustering algorithm for FPGAs, T-VPack [8], uses a greedy approach that selects the least-cost LUTs to pack into a CLB. The cost of a LUT is determined by its criticality and the number of connections shared with LUTs in a cluster; LUTs on the critical path or LUTs that share many connections are prioritized.

Low-power clustering operates on a similar principle as low-power technology mapping. In technology mapping the goal is to cover high activity gate-to-gate connections within the same LUT to hide the switching power of these nodes. In

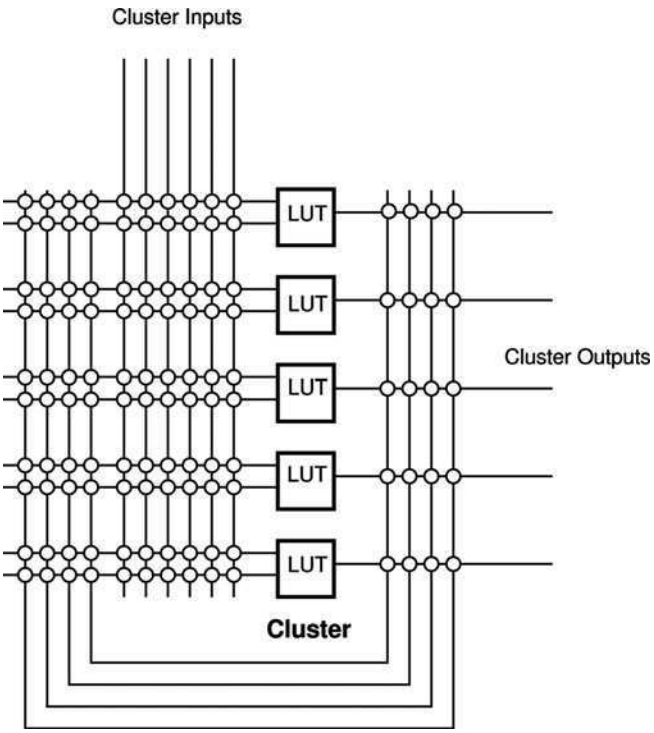


Fig. 10.6 CLB Local interconnect

low-power clustering, the goal is to place high activity LUT to LUT connections within the same CLB, utilizing the more energy efficient local interconnect. This can be accomplished by modifying the cost function of T-VPack to assign minimum cost to the LUTs with the highest activity inputs and outputs [33]. Lamoureux et al. demonstrated that for a cluster size of 4, an average energy reduction of 12.6% is achievable.

An alternative technique to reduce power through improved clustering is to depopulate CLBs, or remove LUTs from fully occupied CLBs and place them elsewhere. Depopulation has been shown to reduce routing resource demand, decreasing both average wirelength and switch utilization [15]. Approximately 13% total power can be saved through depopulation-based clustering [49].

10.3.3 LUT Input Transformations

Several leakage current paths exist in pass transistor multiplexer-based LUTs (Fig. 10.2a). The magnitude of leakage current flowing through these paths depends

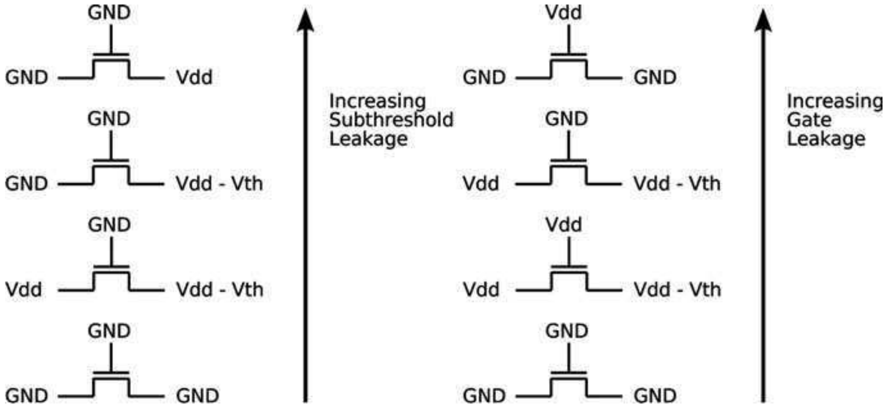


Fig. 10.7 Leakage paths in NMOS pass transistors [22]. (**a**) Subthreshold leakage; (**b**) Gate Leakage

heavily on the voltages on each of the pass transistor nodes. Gate leakage and subthreshold leakage are the two primary sources of leakage power. Figure 10.7 shows different voltage configurations of NMOS pass transistors and how they rank in terms of both subthreshold and gate leakage. We see that gate leakage is maximized when V_{GS} is large and V_{DS} is small, while subthreshold leakage is maximized with large V_{DS}. For most technologies subthreshold leakage dominates gate leakage, so large V_{DS} voltages should be avoided.

If we consider pairs of connected pass transistors in the LUT multiplexer (e.g., the transistors connected SRAM-00 and SRAM-01 in Fig. 10.2a), we see that leakage can be minimized by configuring the LUT so that inputs to pairs of pass transistors are either both “0” or “1” (which minimizes V_{DS} for both transistors). This information can be exploited by using an algorithm to rearrange the LUT inputs such that the LUT is configured for minimal leakage [6, 22]. The algorithm examines all 2^k input permutations and determines which permutation pairs together the most “0” and “1” configuration bits. Through this technique an average of 50.3% leakage energy can be saved.

10.3.4 Placement

The next step after obtaining a clustered CLB netlist is to place the netlist on a physical FPGA architecture. Placement algorithms have traditionally tried to minimize delay by placing critical CLBs close together, reducing the amount of capacitive interconnect a net must pass through. Low-power placement algorithms attempt to place CLBs with high activity connections between them close together. Similar to clustering, the goal is to minimize the amount of activated general-purpose interconnect – shorter nets will charge and discharge less interconnect capacitance, which is desirable for highly active nets. Several works have implemented simulated annealing [28] placers with cost functions modified to reflect switching activity [21, 33, 53]. Each of these placers calculate the cost of a swap by adding a power term to the existing wirelength and delay terms. The power term includes the switching activity of the nets involved in the swap along with their capacitance. However, because net capacitance is not yet known it must be estimated; [53] and [21] include more accurate methods of capacitance estimation that lead to improved results.

Low-power placers must be careful to balance the intents of minimizing wirelength of critical nets and minimizing wirelength of highly active nets. Attempting to place circuits using low-power annealers may increase circuit delay for designs with many critical, low-activity nets. On average, placers without capacitance estimation reduce total power by 3.0% with 4% delay cost; placers with capacitance estimation can achieve reductions of 8.6–13% with delay cost of 1–3% [21, 53].

10.3.5 Routing

The final FPGA CAD step connects placed CLBs together via the programmable routing fabric; FPGA routers typically use the PathFinder algorithm [45]. Nets are routed one by one through the routing fabric using a least-cost path algorithm and are iteratively rerouted until convergence conditions are met. Nets are allowed to share resources so long as those resources are on the least-cost path. The cost of a routing resource (e.g., switch, wire segment) is calculated using the delay of that resource, the criticality of the net currently being routed, and the number of nets currently sharing the resource. Over time the cost of sharing increases rapidly, helping to negotiate congestion and separate the nets.

PathFinder can be modified for low-power routing by adjusting the cost function to account for net activity and capacitance [21, 33], similar to the placement case. The cost of a resource is modified to include the activity of the net being routed and the capacitance of the candidate routing resource. In modern FPGA architectures, routing resources are not identical and have differing capacitances; hence, lower capacitance paths can be selected. Ideally, high activity nets should be mapped to low capacitance routes. However, as in placement, power-aware routers must be careful to balance delay and power minimization; low activity routes may end up being critical. In fact, power-aware routers have been shown to reduce total power by 2.6% but with a delay increase of 3.8%.

10.3.6 Glitch Reduction

A glitch is a switching event in a combinational circuit that does not contribute to the computation of that circuit's final value. Glitches occur when inputs to a gate arrive at different times; Fig. 10.8 shows an example of a glitch. Glitch energy has been shown to be around 20% [32, 43] of total dynamic energy.

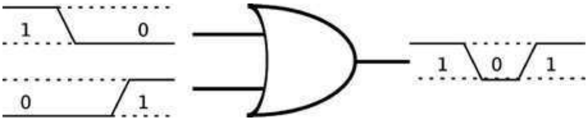


Fig. 10.8 Example of a glitch

Several optimization techniques have been proposed to reduce energy lost to glitches in FPGAs. Wilton et al. proposed pipelining circuits to reduces glitches [54]. A highly pipelined design has fewer glitches because it limits the number of logic levels between registers. Pipelining can reduce energy per operation substantially (40 − 90%) for circuits with significant glitching at a cost of latency. Glitches can also be minimized during CAD steps in either technology mapping [12] or routing [16]. Technology mappers already target reduced switching activity, so only minor modifications need to be made to model glitches. Glitch-aware routers attempt to route early or late arriving signals through alternative paths in the routing fabric such that all signals arrive simultaneously. Glitch-aware CAD has been shown to reduce dynamic power by 11–18.7%.

10.3.7 Full CAD Flows

Lamoureux et al. studied the impact of combining each of these power-aware CAD steps into one uniform low-power CAD flow [33]. They discovered that the power reductions from each of the individual CAD steps were not cumulative. Table 10.2 reviews their results. Technology mapping and clustering yield the most significant power reductions individually of 7.5 and 12.6%, respectively. However, when combined they reduce power by only 17.6%. This is largely due to the fact that

Table 10.2 Energy savings from using low-power CAD tools [33]

Mapping	Clustering	Placement	Routing	Energy savings (%)
Base	Base	Base	Base	0.0
Base	Base	Base	Power	2.6
Base	Base	Power	Base	2.9
Base	Base	Power	Power	5.7
Base	Power	Base	Base	12.6
Base	Power	Base	Power	14.8
Base	Power	Power	Base	15.9
Base	Power	Power	Power	17.9
Power	Base	Base	Base	7.5
Power	Base	Base	Power	9.9
Power	Base	Power	Base	10.1
Power	Base	Power	Power	12.6
Power	Power	Base	Base	17.6
Power	Power	Base	Power	19.5
Power	Power	Power	Base	20.6
Power	Power	Power	Power	22.6

power-aware technology mapping already hides many high activity nodes within LUTs, so power-aware clustering has fewer high activity nodes to optimize by clustering together. Place and route individually save 2.6 and 2.9% energy; when combined with mapping and clustering they add an additional 5.0% rather than 5.6%. For a complete power-aware CAD flow a combined power reduction of 22.6% can be achieved.

10.4 Architecture

While CAD transformations can provide power savings by optimizing the way in which circuits are mapped to FPGAs, changing the physical FPGA architecture can provide significant additional benefits. Architectural changes affect the structure of the FPGA on a global scale by reorganizing the design of logic and interconnect.

10.4.1 Logic Block

FPGA logic is organized into CLBs that contain N k-input LUTs (Fig. 10.1 and 10.6). Values chosen for N and k can impact the overall dynamic power dissipation of an FPGA architecture. Several researchers have explored this design space to determine energy favorable values of k and N [36, 43, 46].

Changing k can have the following tradeoffs in terms of power: first, if k is increased, CLBs require more local routing to connect all LUT input pins (Fig. 10.6), increasing the dynamic energy of the interconnect local to the CLB. However, larger LUTs can implement more complex functions, reducing the total number of LUTs and CLBs. Fewer CLBs mean less demand for global routing resources, saving dynamic interconnect energy. The tradeoff in selecting N is similar: large values of N increase CLB capacity and functionality, which increases local interconnect energy but reduces global interconnect energy. k and N impact leakage energy solely by changing the total area of a design—larger area means more devices leaking. It has been shown that $k = 4$ minimizes area [3] and therefore leakage energy [43].

Lin et al. examined total FPGA energy as a function of k and N [43] (Fig. 10.9). They found that a LUT input size of $k = 4$ minimizes total energy, and that selecting either $k = 3$ or $k = 5$ can increase energy dissipation by as much as 50%. They also determined that smaller cluster sizes reduce energy but with diminishing returns; for $k = 4$ cluster sizes from $N = 6 - 10$ typically use the least energy.

10.4.2 Interconnect Topology

Because interconnect power dominates total FPGA power (Fig. 10.3), it is important to select an appropriate interconnect topology and configuration. FPGA global interconnect is typically parameterized by the length of a wire segment, the

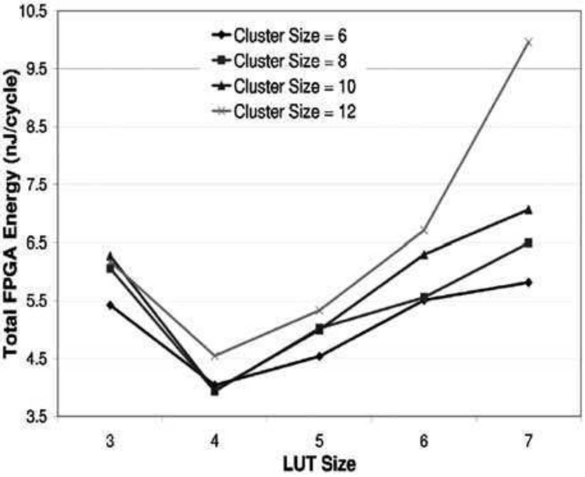


Fig. 10.9 FPGA energy as a function of LUT input size (*k*) and cluster size (*N*) [43]

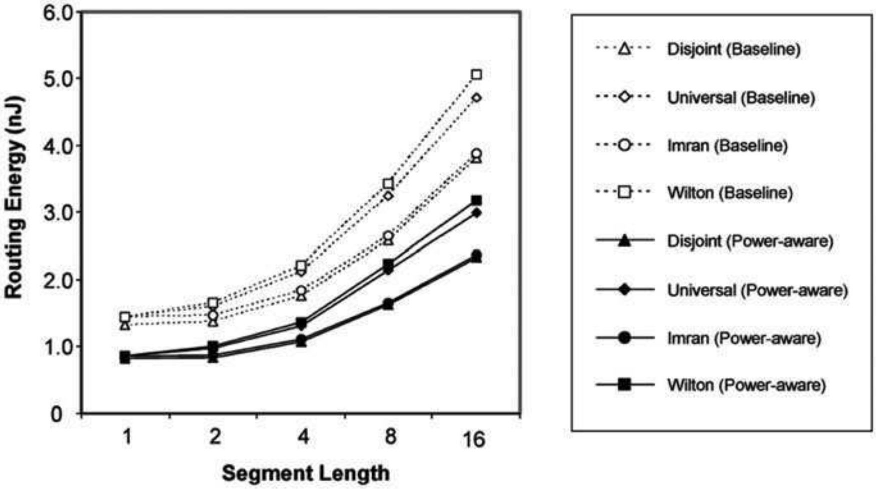


Fig. 10.10 FPGA energy as a function of segment length and switchbox topology [46]

connectivity of switchboxes, and the directionality of the segment (bi-directional or directional).

Poon et al. [46] studied the impact of segment length and switchbox configurations on FPGA energy (Fig. 10.10). A longer wire segment connects to more switches, increasing its capacitance; however, longer segments should lead to fewer total switches. They found the shortest segments $L_{seg} = 1$ and disjoint style switchboxes are the most energy efficient.

In the past, FPGAs were manufactured with bi-directional switches. These switches suffer from the inefficiency that, once configured, an FPGA only uses a

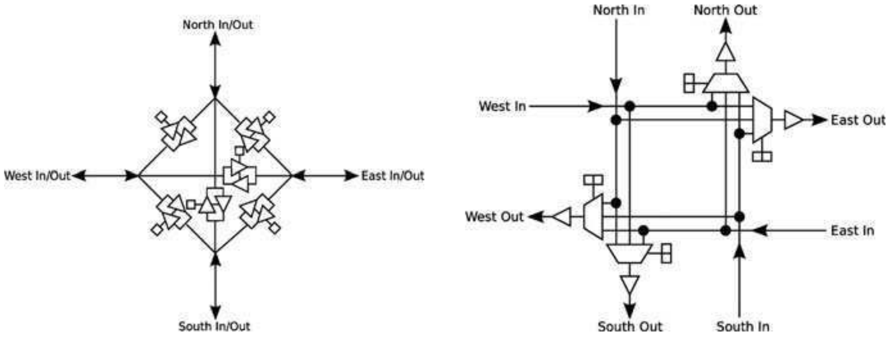


Fig. 10.11 FPGA switches [34]. (**a**) Bi-directional (**b**) Directional

switch in one direction; hence, only 50% of drivers are ever utilized. Directional switches drive segments in a single direction, ensuring that all drivers could be utilized. Figure 10.11 compares the two switch types. Figure 10.2b depicted a single direction of a bidirectional switch; two of these switches are connected in a loop to create a bidirectional switch as shown in Fig. 10.11a. The multiplexer in the directional switch can be implemented with either pass transistor or static logic. While directional drivers would seem to require more wiring because wires can only be utilized in one direction, in practice they use the same number of wires as the bi-directional case, saving area and improving delay [34]. Jamieson et al. [26] studied the effect of directional versus bi-directional switches on interconnect energy, and found that directional wires also save significant energy.

10.4.3 Dynamic Voltage Scaling

A technique commonly used in processors to reduce dynamic energy is dynamic voltage scaling (DVS) (Chapter 2, section 2.1). DVS reduces V_{dd}, saving both dynamic and static energy (Equations (10.1) and (10.2)) but at the expense of increasing delay. DVS is useful in scenarios where a design needs to operate at a target frequency. In these cases, V_{dd} can be lowered to the point where the target is still achieved, minimizing the energy wasted on margins. The exact value of V_{dd} can be different between chips due to variation and can change over time due to environmental variation; hence, an on-chip control circuit with delay feedback is typically used to adjust V_{dd}.

DVS in FPGAs was examined by [13]. They use a design-specific measurement circuit that tracks the delay of a design's critical path to provide feedback to the voltage controller. Figure 10.12 shows the control and measurement circuit; for measurement, a chain of 128 inverters is constructed with identically clocked flip-flops on each inverter output. In a single clock cycle the input to the inverter chain will propagate a specific distance dependent on the delays of the inverters due to environmental conditions. The distance through the chain is set to be identical to the critical path delay in a nominal environment, and it is assumed that the measurement circuit

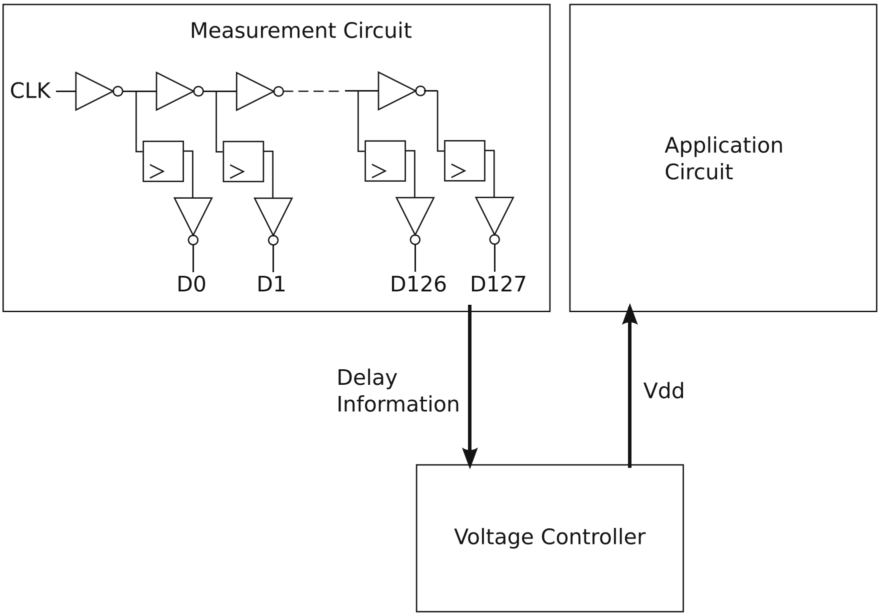


Fig. 10.12 Dynamic voltage scaling for FPGAs [13]

tracks the critical path delay under environmental variation. Through this technique energy savings of 4–54% can be observed.

The correctness of this design depends critically on how well matched the reference circuit chain is to the circuit critical path. More robust techniques will use the path itself as the reference for tuning voltage. The most robust designs augment the registers in the architecture to help monitor timing [7, 9].

10.4.4 Power Gating

Leakage power is a significant source of FPGA power dissipation, and is expected to increase as technology scales [51]. As previously discussed, FPGAs use a significant amount of area to provide programmability. In addition, to be widely usable FPGAs over-provision resources so that customers can map large and complex designs. As a consequence significant portions of an FPGA are often unused. Instead of leaving these unused portions to sit idle and leak, it is possible to use power gating (Chapter 2, section 2.4.5) to disconnect the power supply from these regions and eliminate leakage of unused devices.

Figure 10.13 shows a basic power gating circuit. To support power gating a high V_{th} sleep transistor must be inserted between the power supply and the block to be gated. The high threshold ensures that leakage through the sleep transistor will be negligible. The gate of the transistor is tied to a control bit that can either be set at

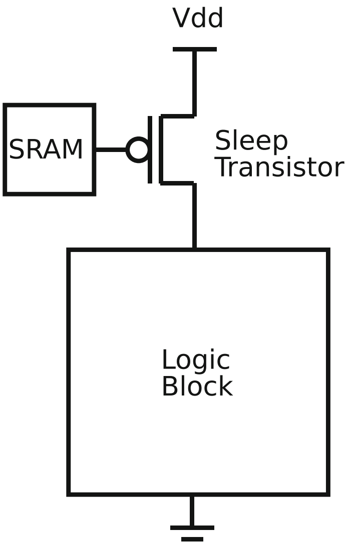


Fig. 10.13 Power gating circuitry

configuration time (i.e., static power gating) or during runtime (i.e., dynamic power gating).

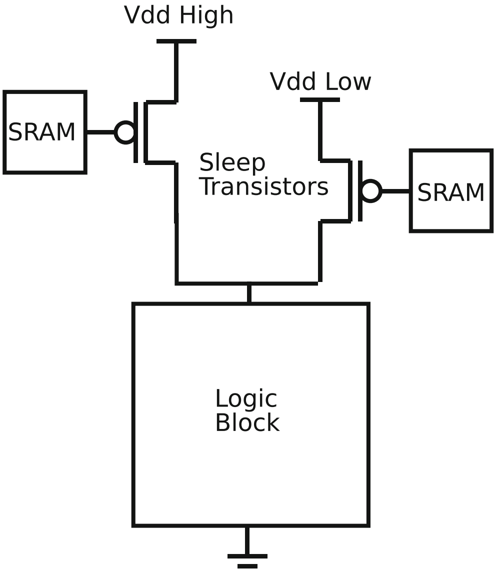
Sleep transistors increase area and also add delay to the gated block. Sizing up the transistors negates the speed penalty, but it also increases the area penalty and reduces the effectiveness of leakage control (wider transistors leak more). Another tradeoff in applying power gating is selecting the appropriate block size to gate. Gating off smaller blocks yields more coverage in disabling unused devices and increases power savings, but at the cost of using many sleep transistors. Gating off larger blocks amortizes the cost of the sleep transistors but makes it difficult to disable the largest number of unused devices.

Many researchers have explored different points in the design space for power gating granularity in FPGAs. Calhoun et al. [10] perform power gating at the gate level; Gayasen et al. [19] use larger power gating blocks, choosing to gate off regions of four CLBs at a time. To enhance leakage reduction they develop a region-constrained placement algorithm to increase the number of regions that can be power gated. They show 20% leakage power savings and that coarse-grained power gating with improved placement achieves the same results as fine-grained power gating. Rahman et al. [47] suggest that a combination of fine and coarse-grained power gating provides the best results.

10.4.5 Dual V_{dd}

Instead of simply using sleep transistors to disable or enable blocks of logic, they can be used to select from different supply voltages (V_{dd}) to power the block. Figure 10.14 shows the basic circuit for providing dual supply voltages. Reducing V_{dd} saves dynamic and static energy, but at the cost of performance. However, not all

Fig. 10.14 Dual V_{dd} design



paths are critical, and hence only elements on critical paths need to be placed in high V_{dd} regions. Paths with timing slack can travel through low V_{dd} blocks. Identifying those elements and assigning an appropriate V_{dd} are therefore important for dual V_{dd} designs. Level conversion is necessary when moving from a low V_{dd} region to a high V_{dd} region, and level converters add delay, area, and energy overhead.

Dual V_{dd} design has been studied extensively in the FPGA literature [5, 18, 25, 37, 38, 44]. Early work placed level converters at the inputs to each wire segment [37, 38], but later work [44] demonstrated that this placement costs significant area and leakage. More recent work has developed techniques for placing level converters only at the inputs to CLBs and developed V_{dd} assignment algorithms to guarantee that no high V_{dd} switch follows a low V_{dd} switch [25]. Furthermore, transformations can be performed on the netlist to distribute timing slack to all nets such that total power is minimized. Dual V_{dd} FPGA architectures typically achieve $\approx 50\%$ power savings.

10.4.6 Dual V_{th} and Body Biasing

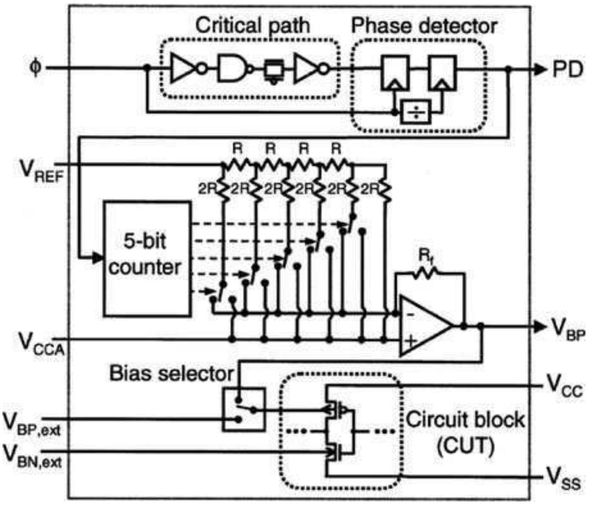
Similar to dual V_{dd} architectures for reducing total power, dual V_{th} and body-biased FPGAs have been proposed to reduce leakage power. Dual V_{th} FPGAs define low and high V_{th} regions at fabrication time. High V_{th} regions reduce leakage at the cost of increased delay. Body-biased FPGAs use embedded circuitry to change the body to source voltage for regions at configuration time. The effect of changing the body to source voltage V_{bs} on V_{th} is

$$V_{\mathrm{th}} = V_{\mathrm{th0}} + \gamma \left(\sqrt{|\Phi_{\mathrm{s}} - V_{\mathrm{bs}}|} - \sqrt{|\Phi_{\mathrm{s}}|} \right) \tag{10.3}$$

Where V_{th0} is the ideal V_{th} at $V_{bs} = 0$, γ is the body bias coefficient, and Φ_s is the surface potential. Hence, by applying negative values of V_{bs} to a region its V_{th} can be increased.

Figure 10.15 shows the circuitry required for body biasing. To determine the correct bias voltage, a representative critical path from the circuit to be biased is replicated and its delay is compared to a reference clock using a phase detector. The output of the phase detector is fed into a digital counter, which is converted into an analog signal using a R–2R resistor network and op-amp. This signal is then used to select the body bias voltage for the target circuit block.

Fig. 10.15 Body bias circuit [50]



Much work has also been done in dual V_{th} design for FPGAs [24, 27, 39, 52] with body biasing being integrated into commercial FPGAs [35]. Similar tradeoffs exist when mapping circuits to either dual V_{th}/body biased architectures or dual V_{dd} architectures. Critical paths must be placed on low V_{th} blocks to ensure minimal delay reduction, and timing slack must be available to save leakage energy on non-critical paths. Block granularity is important [24] and selection of approriate body bias voltages is important [27].

While the previous discussion of V_{th} and V_{dd} selection applies at the level of the mapped design, it is also profitable to use different V_{th} devices for circuits that play different roles in the FPGA architecture. Notably, high V_{th} devices can significantly reduce the configuration SRAM bit leakage reduced without impacting area or delay [52]. Configuration bits are a prime candidate for high V_{th} transistors because they constitute a significant fraction of FPGA area and are always on and leaking. Fortunately, configuration bits are set only at configuration time and do not contribute any delay or switching energy to mapped circuits. Increasing configuration SRAM V_{th} can reduce total leakage energy by up to 43%. Today's commercial FPGAs are fabricated with three different threshold voltages [29].

10.5 Circuits

Below the level of architectural techniques are circuit optimizations. These techniques typically attempt to redesign LUT and interconnect switch circuits to reduce both static and dynamic power.

10.5.1 Low Swing Interconnect

Reducing interconnect capacitance is one way to reduce the dynamic energy dissipated by FPGA switches. However, capacitance is only a linear term in dynamic energy (Equation (10.1)); V_{dd} is quadratic. Hence, greater energy savings could be achieved by reducing the voltage switched in the interconnect.

A low swing interconnect segment consists of a driver and receiver operating at nominal voltages, with the wire between them operating at reduced voltage. The driver converts a full swing input into a low swing interconnect signal and the receiver converts it back. With this technique the amount of dynamic energy dissipated in interconnect segments can be reduced significantly. A common drawback of low swing interconnect is the slow speed of the receiver circuit. Figure 10.16 shows a low swing segment that uses cascode and differential circuitry to improve receiver speed [20]. By employing this technique throughout an FPGA, interconnect energy can be reduced by a factor of 2.

10.5.2 Glitch Reduction

Section 10.3.6 introduced techniques to transform the design to reduce glitches. With circuit-level support, we can introduce additional opportunities to reduce glitches.

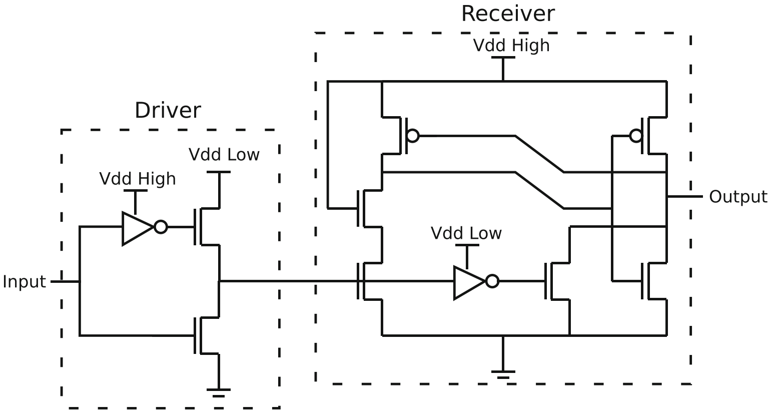


Fig. 10.16 Low swing interconnect circuit [20]

GlitchLess is a circuit level technique to eliminate glitches [32]. GlitchLess inserts programmable delay elements at the inputs of each logic element, and programs them, post-fabrication, such that arriving signals are aligned. Figure 10.8 showed an example of a glitch; Fig. 10.17 shows how to eliminate this glitch by adding a delay element. On average this eliminates 87% of glitches, reducing overall FPGA power by 17%.

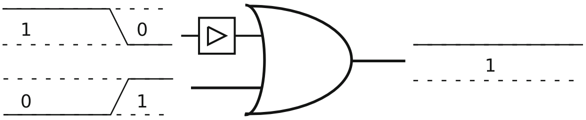


Fig. 10.17 Removing glitches through programmable delay elements

10.6 Devices and Technologies

The lowest level at which FPGA designers can attempt to reduce power consumption is at the device and technology level. Fundamental advances in several emerging technologies have made is possible to build FPGAs out of more than just conventional CMOS transistors. We briefly examine a few technologies where studies suggest the 3D ICs may reduce FPGA power consumption.

10.6.1 Three-Dimensional Integration

A three-dimensional integrated circuit (3D IC) is a chip with two or more active layers of devices. 3D ICs have two distinct physical advantages over two-dimensional integrated circuits (2D ICs). First, they can achieve significantly higher logic density from adding devices in a third dimension. Second, average wirelength decreases because logic elements effectively move closer together. A reduction in average wirelength means a reduction in interconnect capacitance, which is the dominant source of FPGA power.

There are three ways to stack 3D ICs: wafer-to-wafer, die-to-die, and monolithic. Wafer-to-wafer and die-to-die methods manufacture wafers and dies separately and stack them together, connecting layers using through-silicon vias (TSVs). The primary challenge with these two techniques is that TSVs are generally very large and only limited connections can be made between layers. Monolithic stacking directly fabricates active layers of silicon together, which results in much higher inter-layer connection density. The primary challenge with monolithic stacking is that the temperature needed to create an active layer can damage layers and wiring beneath it.

Ababei et al. explored the benefits of a die-to-die 3D FPGA architecture [1]. They developed an optimized 3D place-and-route tools to aid in wirelength reduction.

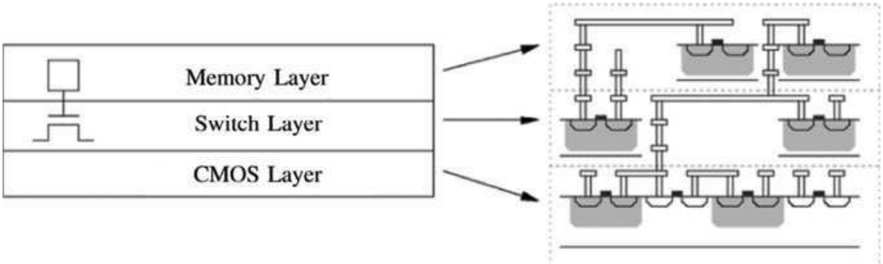


Fig. 10.18 Monolithic 3D FPGA [41]

Assuming a ten layer device they demonstrated that a die stacked 3D FPGA benefits from a 25% average wirelength reduction.

Lin et al. studied a monolithically stacked FPGA [41, 42] where logic, routing, and configuration all occupy independent layers (Fig. 10.18). This is possible, in part, because configuration memories make up a significant fraction of the area in FPGAs. The configuration memories do not require high-speed logic transistors, so can use alternate device technologies that do not require high temperature processing. Removing the memory area from the logic reduces one of the large components of area overhead that drive longer wires and hence larger capacitance in FPGA designs compared to ASICs. They demonstrate that a 3D monolithic FPGA can reduce area, delay, and dynamic power by 3.3×, 2.51×, and 2.93×, respectively.

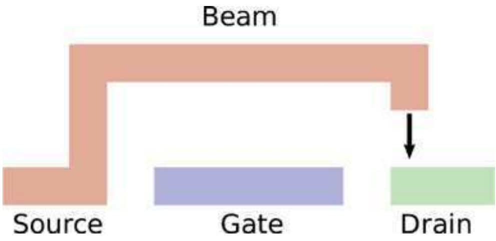
The main challenge in 3D ICs is manufacturing; however, an important secondary challenge is cooling. ICs are typically cooled on the package surface, and with conventional cooling techniques (e.g., heatsinks and fans) power density is limited to approximately 100 W/cm^2. 3D ICs have significantly higher power densities than 2D chips because of many more active transistors per unit surface area. While 3D ICs reduce wire capacitance such that overall power efficiency is improved, it may prove difficult to provide sufficient cooling proportional to the increase in power density.

10.6.2 Nanoelectromechanical Relays

Nanoelectromechanical (NEM) relays are electrostatically actuated mechanical switches that are excellent candidates to replace FPGA interconnect switches. NEM relays hold their on/off state (i.e., they are hysteretic), enabling them to replace both the NMOS pass transistor and SRAM configuration bit in a conventional FPGA switch with a single device. They have much lower on resistance than NMOS pass transistors, reducing delay. In terms of power, NEM relays exhibit zero leakage current. Hence, significant reductions in leakage power are possible.

NEM relays can be made out of conventional CMOS processing materials (e.g., silicon, metal) or carbon nanotubes, which are more difficult to integrate into a CMOS manufacturing flow. Figure 10.19 shows a three-terminal relay. The relay consists of source, gate, and drain electrodes and a mechanical deflecting beam.

Fig. 10.19 NEMS switch



When a voltage V_{gs} is applied to the switch, electrostatic force attracts the beam downward toward the gate. An opposing, intrinsic elastic force in the beam resists the deflection. When V_{gs} is greater than the pull-in voltage $\mathrm{V_{pi}}$, the beam collapses and makes contact with the drain, enabling conduction between the source and drain. To disconnect the beam, Vgs is lowered below the pull-out voltage $\mathrm{V_{po}}$. Because $\mathrm{V_{po}} < \mathrm{V_{pi}}$, the switch exhibits hysteresis with an operating range between $\mathrm{V_{po}}$ and $\mathrm{V_{pi}}$.

Zhou et al. examined replacing switches in LUTs with carbon nanotube NEM relays, demonstrating a 91% reduction in LUT power [55]. Chen et al. explored conventional metal NEM relays as a replacement for interconnect switches [11]. They found that NEM relays can reduce area, delay, and leakage power by 43.6, 28, and 37%, respectively.

The main challenges in using NEM relays are integrating with a conventional CMOS manufacturing process and ensuring reliable mechanical contacts over many switching events. The deflecting beam can suffer from stiction (failure to release after deflection), tip bouncing, and other mechanical failures.

10.7 Summary

Table 10.3 summarizes the techniques covered in this chapter, the level at which they operate, the terms in the power equations they attempt to reduce (from Equations (10.1) and (10.2)), the type of power reduction, and the demonstrated benefit. It is important to note that these benefits will not all be additive.

While FPGAs dissipate considerably more power than ASICs when fabricated in the same process and solving identical problems, a substantial amount of research in low-power FPGA design in the last decade has begun to narrow this gap, making it easier for differences in technology and problem solved to result in lower energy for the FPGA than the ASIC. This chapter summarized the progress in reducing FPGA power across all levels of design, from CAD and architecture down to circuits and devices, and all components of the FPGA architecture.

Demand for low-power devices will only increase in the future as mobile devices become even more ubiquitous. As technology scales, power limitations will constrain the performance and growth of CPUs, which are already at a power disadvantage to FPGAs. ASICs are declining in popularity due to enormous design and fabrication costs, which will only increase. FPGAs have the unique opportunity to

Table 10.3 Roundup of low-power FPGA techniques

Technique	Level	Reduces	Benefit type	Benefit(%)
Technology mapping	CAD	C, α	P_{total}	7.6–17
Clustering	CAD	C, α	P_{total}	13
LUT input transformation	CAD	I_{leak}	P_{static}	50
Placement	CAD	C, α	P_{total}	3.0–13
Routing	CAD	C, α	P_{total}	3
Glitch routing/placement	CAD	α	P_{dynamic}	11–19
Logic block architecture	Architecture	C	P_{total}	48
Interconnect architecture	Architecture	C	P_{total}	12
Dynamic voltage scaling	Architecture	V_{dd}	P_{total}	4–54
Power gating	Architecture	I_{leak}	P_{static}	20
Dual V_{dd}	Architecture	V_{dd}	P_{total}	50
Dual V_{th}/body biasing	Architecture	I_{leak}	P_{static}	43
Low swing interconnect	Circuits	V_{dd}	P_{total}	50
Glitchless	Circuits	α	P_{total}	17
3D integration	Devices	C	P_{dynamic}	66
NEM relays	Devices	I_{leak}	P_{static}	37

become the low-power devices of future by continuing the close the power gap with innovations in power reduction. Chapter 12 shows how efficient handling of variation and wear can further narrow or reverse the energy gap and increasingly favor more FPGA-like architectures.

References

1. Ababei C, Mogal H, Bazargan K (2006) Three-dimensional place and route for FPGAs. IEEE Trans Comput Aided Des Integr Circ Syst 25(6):1132–1140
2. Abnous A, Zhang H, Wan M, Varghese G, Prabhu V, Rabaey J (eds) (2002) The application of programmable DSPs in mobile communications, Chichester, UK. doi: 10.1002/0470845902 chap. 17: The pleiades architecture. Wiley, pp 327–359
3. Ahmed E, Rose J (2000) The effect of LUT and cluster size on deep-submicron FPGA performance and density. In: Proceedings of the international symposium on field-programmable gate arrays. ACM, New York, NY, pp 3–12
4. Anderson J, Najm F (2002) Power-aware technology mapping for LUT-based FPGAs. In: Proceedings of the international conference on field-programmable technology, pp 211–218
5. Anderson J, Najm F (2004) Low-power programmable routing circuitry for FPGAs. In: Proceedings of the international conference on computer-aided design
6. Anderson J, Najm F, Tuan T (2004) Active leakage power optimization for FPGAs. In: Proceedings of the international symposium on field-programmable gate arrays
7. Austin T, Blaauw D, Mudge T, Flautner K (2004) Making typical silicon matter with Razor. IEEE Comput 37(3):57–65
8. Betz V (1999) VPR and T-VPack: versatile packing, placement and routing for FPGAs. <http://www.eecg.toronto.edu/~vaughn/vpr/vpr.html>. Version 4.30
9. Bowman KA, Tschanz JW, Kim NS, Lee JC, Wilkerson CB, Lu SLL, Karnik T, De VK (2009) Energy-efficient and metastability-immune resilient circuits for dynamic variation tolerance. IEEE J Solid State Circuits 44(1):49–63

10. Calhoun B, Honore F, Chandrakasan A (2003) Design methodology for fine-grained leakage control in MTCMOS. In: Proceedings of the international symposium on low power electronics and design
11. Chen C, Wong HSP, Mitra S, Parsa R, Patil N, Chong S, Akarvardar K, Provine J, Lewis D, Watt J, Howe RT (2010) Efficient FPGAs using nanoelectromechanical relays. In: Proceedings of the international symposium on field-programmable gate arrays. ACM Press, New York, NY, p 273
12. Cheng L, Chen D, Wong MD (2007) GlitchMap: An FPGA technology mapper for low power considering glitches. In: Proceedings of the ACM/IEEE design automation conference, pp 318–323
13. Chow C, Tsui L, Leong P, Luk W, Wilton S (2005) Dynamic voltage scaling for commercial FPGAs. Proceedings of the international conference on field-programmable technology, pp 173–180
14. DeHon A (1998) Comparing computing machines. In: Configurable computing: technology and applications, proceedings of SPIE, vol. 3526, SPIE
15. DeHon A (1999) Balancing Interconnect and computation in a reconfigurable computing array (or, why you don't really want 100% LUT utilization). In: Proceedings of the international symposium on field-programmable gate arrays, pp 69–78
16. Dinh Q, Chen D, Wong M (2010) A routing approach to reduce glitches in low power FPGAs. IEEE Trans Comput Aided Des Integr Circ Syst 29(2):235–240
17. Farrahi AH, Sarrafzadeh M (1994) FPGA technology mapping for power minimization. In: Proceedings of the international conference on field-programmable logic and applications, Springer, London, pp 66–77
18. Gayasen A, Lee K, Vijaykrishnan N, Kandemir M, Irwin M, Tuan T (2004) A Dual-Vdd low power FPGA architecture. In: Proceedings of the international conference on field-programmable logic and applications, Springer, pp 145–157
19. Gayasen A, Tsai Y, Vijaykrishnan N, Kandemir M, Irwin MJ, Tuan T (2004) Reducing leakage energy in FPGAs using region-constrained placement. In: Proceedings of the international symposium on field-programmable gate arrays, pp 51–58
20. George V, Zhang H, Rabaey J (1999) The design of a low energy FPGA. In: Proceedings of the international symposium on low power electronics and design, pp 188–193
21. Gupta S, Anderson J, Farragher L, Wang Q (2007) CAD techniques for power optimization in Virtex-5 FPGAs. In: Proceedings of the IEEE custom integrated circuits conference, pp 85–88
22. Hassan H, Anis M, Elmasry M (2008) Input vector reordering for leakage power reduction in FPGAs. IEEE Trans Comput Aided Des Integr Circ Syst 27(9):1555–1564
23. Hauck S, DeHon A (eds) (2008) Reconfigurable computing: the theory and practice of FPGABased computation. systems-on-silicon. Elsevier Burlington, MA
24. Hioki M, Kawanami T, Tsutsumi T, Nakagawa T, Sekigawa T, Koike H (2006) Evaluation of granularity on threshold voltage control in flex power FPGA. In: Proceedings of the international conference on field-programmable technology, pp 17–24
25. Hu Y, Lin Y, He L, Tuan T (2008) Physical synthesis for FPGA interconnect power reduction by dual-Vdd budgeting and retiming. ACM Trans Des Autom Electron Syst 13(2):1–29
26. Jamieson P, Luk W, Wilton SJ, Constantinides GA (2009) An energy and power consumption analysis of FPGA routing architectures. In: Proceedings of the international conference on field- programmable technology, pp 324–327
27. Kawanami T, Hioki M, Matsumoto Y, Tsutsumi T, Nakagawa T, Sekigawa T, Koike H (2006) Optimal set of body bias voltages for an FPGA with field-programmable Vth components. Proceedings of the international conference on field-programmable technology, pp 329–332
28. Kirkpatrik S, Gellatt Jr, CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
29. Klein M (2005) The Virtex-4 power play. Xcell J (52):16–19
30. Kuon I, Rose J (2007) Measuring the gap between FPGAs and ASICs. IEEE Trans Comput Aided Des Integr Circ Syst 26(2):203–215

31. Kusse E, Rabaey J (1998) Low-energy embedded FPGA structures. In: Proceedings of the international symposium on low power electronics and design, pp 155–160
32. Lamoureux J, Lemieux GGF, Wilton SJE (2008) GlitchLess: dynamic power minimization in FPGAs through edge alignment and glitch filtering. IEEE Trans VLSI Syst 16(11):1521–1534
33. Lamoureux J, Wilton S (2003) On the interaction between power-aware FPGA CAD algorithms. In: Proceedings of the international conference on computer-aided design. IEEE Computer Society, Washington, DC
34. Lemieux G, Lee E, Tom M, Yu A (2004) Directional and single-driver wires in fpga interconnect. In: Proceedings of the international conference on field-programmable technology, pp 41–48
35. Lewis D, Ahmed E, Cashman D, Vanderhoek T, Lane C, Lee A, Pan P (2009) Architectural enhancements in Stratix-III and Stratix-IV. In: Proceedings of the international symposium on field-programmable gate arrays, ACM, pp 33–42
36. Li F, Chen D, He L, Cong J (2003) Architecture evaluation for power-efficient FPGAs. In: Proceedings of the International symposium on field-programmable gate arrays, ACM Press, New York, NY, p 175
37. Li F, Lin Y, He L (2004) FPGA power reduction using configurable dual-Vdd. In: Proceedings of the ACM/IEEE design automation conference, ACM, pp 735–740
38. Li F, Lin Y, He L (2004) Vdd programmability to reduce FPGA interconnect power. In: Proceedings of the International Conference on Computer-Aided Design, IEEE, pp 760–765
39. Li F, Lin Y, He L, Cong J (2004) Low-power FPGA using pre-defined dual-Vdd/dual-Vt fabrics. In: Proceedings of the international symposium on field-programmable gate arrays, ACM, p 4250
40. Li H, Katkoori S, Mak WK (2004) Power minimization algorithms for LUT-based FPGA technology mapping. ACM Trans Des Autom Electron Syst 9(1):33–51
41. Lin M, El Gamal A (2009) A low-power field-programmable gate array routing fabric. IEEE Trans VLSI Syst 17(10):1481–1494
42. Lin M, El Gamal A, Lu YC, Wong S (2007) Performance benefits of monolithically stacked 3- D FPGA. IEEE Trans Comput Aided Des Integr Circ Syst 26(2):216–229
43. Lin Y, Cong J (2005) Power modeling and characteristics of field programmable gate arrays. IEEE Trans Comput Aided Des Integr Circ Syst 24(11):1712–1724
44. Lin Y, He L (2006) Dual-Vdd interconnect with chip-level time slack allocation for FPGA power reduction. IEEE Trans Comput Aided Des Integr Circ Syst 25(10):2023–2034
45. McMurchie L, Ebeling C (1995) PathFinder: a negotiation-based performance-driven router for FPGAs. In: Proceedings of the international symposium on field-programmable gate arrays, ACM, pp 111–117
46. Poon K, Wilton S, Yan A (2005) A detailed power model for field-programmable gate arrays. ACM Trans Des Autom Electron Syst 10:279–302
47. Rahman A, Das S, Tuan T, Trimberger S (2006) Determination of power gating granularity for FPGA fabric. In: Proceedings of the IEEE custom integrated circuits conference, pp 9–12
48. Shang L, Kaviani A, Bathala K (2002) Dynamic power consumption in Virtex-II FPGA family. In: Proceedings of the international symposium on field-programmable gate arrays, ACM, p 164
49. Singh A, Marek-Sadowska M (2002) Efficient circuit clustering for area and power reduction in FPGAs. In: Proceedings of the international symposium on field-programmable gate arrays, ACM, New York, NY, pp 59–66
50. Tschanz J, Kao J, Narendra S, Nair R, Antoniadis D, Chandrakasan A, De V (2002) Adaptive body bias for reducing impacts of die-to-die and within-die parameter variations on microprocessor frequency and leakage. IEEE J Solid State Circuits 37(11):1396–1402
51. Tuan T, Lai B (2003) Leakage power analysis of a 90 nm FPGA. In: Proceedings of the IEEE custom integrated circuits conference, IEEE, p 57
52. Tuan T, Rahman A, Das S, Trimberger S, Kao S (2007) A 90-nm low-power FPGA for battery-powered applications. IEEE Trans Comput Aided Des Integr Circ Syst 26(2):296–300

53. Vorwerk K, Raman M, Dunoyer J, Kundu A, Kennings A (2008) A technique for minimizing power during FPGA placement. In: Proceedings of the international conference on field-programmable logic and applications, pp 233–238
54. Wilton S, Ang S, Luk W (2004) The impact of pipelining on energy per operation in fieldprogrammable gate arrays. In: Proceedings of the international conference on field-programmable logic and applications, Springer, pp 719–728
55. Zhou Y, Thekkel S, Bhunia S (2007) Low power FPGA design using hybrid CMOS-NEMS approach. In: Proceedings of the international symposium on low power electronics and design, ACM, p 19

Chapter 11
Variation and Aging Tolerance in FPGAs

Nikil Mehta and André DeHon

Abstract Parameter variation and component aging are becoming a significant problems for all digital circuits including FPGAs. These effects degrade performance, increase power dissipation, and cause permanent faults at manufacturing time and during the lifetime of an FPGA. Several techniques have been developed to tolerate variation and aging in ASICs; FPGA designers have been quick to adopt and customize these strategies. While FPGAs can use many ASIC techniques verbatim, they have a distinct advantage to aid in the development of more innovate solutions: reconfigurability. Reconfigurability gives us the ability to spread wear effects over the chip which is not possible in ASICs. This chapter examines the impact of variation and wear on FPGAs and notes the benefit that can be gained from variation and aging tolerance techniques that operate open-loop.

11.1 Introduction

Like all scaled semiconductor devices, FPGAs are experiencing dramatic increase in process variability. Every manufactured FPGA is unique in terms of its composition of physical parameters (e.g., channel lengths, oxide thickness, and threshold voltages) for each transistor. FPGAs are also subject to the same aging mechanisms that cause ASICs to degrade in performance and fail over time. However, FPGA manufacturers have the unique challenge that the functionality of an FPGA is not fixed at manufacturing time. Users can map any circuit onto an FPGA, and they expect a fully functioning, high performance design, regardless of the parameter variation imposed on the specific chip they select for mapping. Additionally, because every user design is different, every workload will differ in terms of how it stresses individual circuits within the FPGA. Different temperature and usage pro-

N. Mehta (✉)
Department of Computer Science, California Institute of Technology, Pasadena, CA, USA
e-mail: nikil@caltech.edu

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_11,

files result in unique aging trends. Without any techniques to combat variation and aging, to meet the demand that every manufactured chip will function with any user configuration and usage pattern over time, manufacturers pessimistically guard band timing parameters.

Fortunately, FPGAs have two distinct advantages over fixed designs like ASICs and CPUs. First, they have spare resources: user designs infrequently utilize all resources, and even when mapped at full logic capacity, FPGAs still have many additional, unused interconnect resources. These extra resources can be used to avoid resources that have been degraded by parameter variation or aging. Second, FPGAs can reconfigure. There are many ways to map a user's design, and these different mappings can be instantiated over time in the field. When a specific user mapping fails on a particular chip due to parameter variation or aging, another configuration may be tried.

These alternate configurations can be identical mappings for all chips deployed in the field over time, or they can be specific to a component such that every chip receives a unique mapping. Component-specific mapping is a challenging but powerful technique for FPGAs that will be explored in depth in the following chapter. This chapter focuses on traditional, component-oblivious techniques for tolerating variation and aging. Many techniques are adopted directly from ASIC and CPU design flows, but customized for FPGAs and are general in that they can be applied to any digital design without reconfiguration. Some, however, exploit configurability, in a open-loop manner, to combat projected aging. The outline for this chapter is as follows: Section 11.2 reviews how variation impacts FPGAs and Section 11.3 describes techniques for variation tolerance; Sections 11.4 and 11.5 discuss the impact of aging and strategies for lifetime extension.

11.2 Impact of Variation

(Chapter 1) described sources of variation in detail. While die-to-die (D2D) variation has historically been the dominant source of variation, in recent technology nodes within die (WID) variation has increased beyond that of D2D. Systematic, spatially correlated, and random WID variations all have a significant impact on FPGAs; random WID variation in particular has been increasing and will be the dominant source of variation in future technology nodes. Compensating for this random WID variation is a significant challenge as neighboring transistors can have completely different characteristics.

To describe the impact of variation on FPGA designs, metrics like timing yield and power yield are often used. Users of FPGAs typically want all chips containing their mapped designs to achieve a set performance target while staying within a power budget. Timing yield and power yield simply express the percentage of devices that meet those targets. Power yield is often termed leakage yield as static power has begun to dominate total power dissipation.

11.2.1 FPGA-Specific Effects

Variation typically impacts ASICs and FPGAs in similar ways; however, commercial FPGAs do have a few key architectural differences that help to mitigate variation. First, while variation in memory is a serious problem in ASICs and CPUs, it is less of a problem for FPGAs. Memory in FPGAs is predominately used as configuration bits; these bits are typically only written once at power on time and area always on – they are never read or written in timing critical paths. This means that the FPGA can afford to use larger thresholds, supply voltages, and skewed cells (Section 11.3.3) to combat variation without impacting power and performance. Modern FPGAs contain large embedded memories that require similar care as embedded memories in ASICs.

Second, connections between logic elements are longer in FPGAs than in ASICs. Although these connections have larger mean delays, they also have smaller percentage variation. Because FPGA connections must traverse several switches, delay variation averages out over the length of the path. The variation in path delay for a path of N identical Gaussian gates with delay distributions of $(\mu_{\tau_{\text{gate}}}, \sigma_{\tau_{\text{gate}}})$ can be written as

$$\mu_{\tau_{\text{path}}} = N\mu_{\tau_{\text{gate}}} \tag{11.1}$$

$$\sigma^2_{\tau_{\text{path}}} = N\sigma^2_{\tau_{\text{gate}}} \tag{11.2}$$

$$\frac{\sigma_{\tau_{\text{path}}}}{\mu_{\tau_{\text{path}}}} = \frac{1}{\sqrt{N}} \frac{\sigma_{\tau_{\text{gate}}}}{\sigma_{\tau_{\text{gate}}}} \tag{11.3}$$

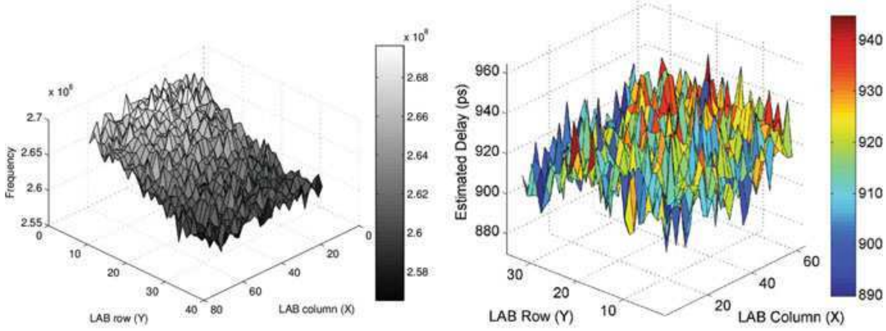
Length N paths have variation reduced by $1/\sqrt{N}$. This means for identical designs, with the way commercial FPGAs are architected (long, unpipelined path lengths), FPGAs may experience less delay variability than ASICs. This can help to improve timing yield.

Another advantage of FPGAs is that they have a small, repeated circuit structure, unlike ASICs which can have a large, arbitrary layout. With the recent explosion of layout design rules, ASIC physical design has become much more difficult. While FPGA manufacturers must also struggle with deep submicron effects, they only have to design, layout, and verify a much smaller structure. Further, because the structure is regular and repeated, it is much easier to understand and potentially avoid the impact of systematic variation. When using optical proximity correction and other resolution enhancement techniques (Chapter 1, Section 1.2.1) to model the effect of diffraction through the mask, a regular, repeated structure means that it is easier to calculate OPC and model its effect on a design.

11.2.2 FPGA-Level Impact

It is difficult to precisely quantify the impact of parameter variation on commercial FPGAs. Because variation data is economically sensitive to vendors, no information characterizing FPGA variability has been officially released. However, several researchers have empirically measured batches of both custom manufactured and commercial FPGAs, characterizing their variation distributions. These studies begin to quantify and document the effect of real variation on FPGAs.

Katsuki et al. measured WID variability for 90-nm custom manufactured FPGAs [2]. To collect variation data they mapped arrays of ring oscillators and measured their frequencies across the chip. Unfortunately, they published data in arbitrary units, so actual delay variation numbers are not available. However, Sedcole et al. measured WID delay variability of 18 commercial 90-nm Altera Cyclone II FPGAs by using a similar ring oscillator measurement technique [10] (Fig. 11.1a). They found that individual logic elements have a random delay variability of $3\sigma/\mu = 3.54\%$; they also found a WID-correlated component of delay variability of 3.66%. Finally, Wong et al. used an at-speed measurement technique to characterize the WID delay variability of both logic and embedded multipliers in a 65-nm Altera Cyclone III FPGA [18] (Fig. 11.1b). They discovered a WID delay variation of $3\sigma/\mu = 5.96\%$. Each of these experiments demonstrate the feasibility of characterizing the delay distribution of individual FPGAs elements. The next chapter will further develop how these measurements can be performed and leveraged to mitigate variation.



(a) Altera Cyclone II [10]

(b) Altera Cyclone III [18]

Fig. 11.1 Measured LUT frequency maps for Altera FPGAs

11.3 FPGA Variation Tolerance

To tolerate FPGA parameter variation, researchers have employed many of the same techniques used for ASICs and CPUs. These methods can operate at the CAD level to statistically account for variability when generating a configuration for a design, or they can operate at the architecture and circuit levels to physically compensate for the effects of variation. Some of these techniques are the same as those used for reducing power consumption but used in a new context.

11.3.1 SSTA CAD

A heavily researched method for dealing with variation during the technology mapping, place and route stages of an ASIC design is the use of statistical static timing analysis (SSTA) (Chapter 4, Section 4.4.3). SSTA attempts to model variability directly in CAD algorithms to help produce circuits that inherently account for variation.

The goal of most CAD algorithms is to minimize the critical path delay of a circuit. Traditional CAD algorithms do not aggressively minimize near-critical paths since their reduction would not reduce the delay of the circuit. However, near-critical paths are important under variation because there is a probability that they will become critical. As expressed in Equations (11.1), (11.2), (11.3), path delays are not constants but distributions. Hence, with some probability, a path may be faster or slower than its nominal value. Near critical paths that have a probability of becoming critical are deemed to be *statistically critical*.

SSTA is a methodology that identifies statistically critical paths and enables CAD algorithms to optimize those paths. Integrating SSTA into clustering, placement and routing algorithms of the FPGA CAD flow simply involves replacing the nominal STA routine with an SSTA routine. The algorithms then proceed as normal, attempting to reduce the delay of statistically critical paths and improve circuit timing yield (e.g., minimize μ_τ and σ_τ of the circuit).

FPGA CAD algorithms modified for SSTA have been studied for placement [5] and routing [3, 11]. Lin et al. studied a full SSTA CAD flow with clustering, placement, and routing and characterized the interaction between each stage [4]. The only modification made to each algorithm is to replace the criticality of a net in all cost functions with its statistical criticality. In clustering, SSTA is only performed at the start of the algorithm; for placement and routing statistical criticalities are updated at the start of every iteration.

Lin et al. [4] examined a 65-nm predictive technology with $3\sigma_{L_{\text{eff}}}/\mu_{L_{\text{eff}}}$ and $3\sigma_{V_{\text{th}}}/\mu_{V_{\text{th}}}$ of 10, 10, and 6% for D2D, WID spatially correlated, and WID random variation. They observed that SSTA-based clustering alone can improve μ_τ and σ_τ by 5.0 and 6.4%, respectively, and improve timing yield by 9.9%. Placement improves μ_τ, σ_τ and timing yield by 4.0, 6.1, and 8.0%; routing achieves 1.4, 0.7, and 2.2% improvement. All three SSTA algorithms combined yield 6.2, 7.5, and 12.6% improvement. As in the power-aware CAD algorithm case from (Chapter 10, Section 10.3.7), they find that individual enhancements to the three stages of FPGA CAD are not additive, and that stochastic clustering provides the majority of benefit.

These benefits come with a 3× runtime increase but with minimal changes to FPGA CAD algorithms. Their primary effect may be to provide a better estimate for the delay of the circuit, reducing some of the conservative margining used to guarantee a target level of performance across almost all chips. Nonetheless, there are several drawbacks to SSTA-based approaches. First, they depend on having accurate models of process variation. Without correct values for the σ, μ, and correlation coefficients of process parameters, SSTA cannot accurately determine which paths are statistically critical. Second, these techniques only improve the probability of a device meeting timing and do not guarantee working parts. SSTA CAD still produces a one-size-fits-all design, and there is non-negligible probability that a particular chip will not yield with a SSTA-enhanced design. Finally, SSTA approaches do not scale well with high random variation, which is expected to dominate future technology nodes. High variation spreads out delay distributions, making many more paths statistically critical. It then becomes difficult for the CAD algorithms to optimize and reduce the delay of many paths at once.

11.3.2 Architecture Optimization

Another strategy for mitigating the impact of parameter variations is to optimize the structure of the FPGA by changing key architectural parameters (Chapter 10, Section 10.4). FPGA logic is parameterized by *N*, the number of LUTs per CLB, and *k*, the number of inputs to a LUT. Routing is parameterized by the switchbox type and segment length L_{seg}. Changing these values can have an impact on both timing and leakage yield.

In terms of logic, larger values of *N* and *k* increase the area, delay, and leakage of individual CLBs which will hurt leakage and timing yield. However, the total number of CLBs and required routing resources may decrease, which would improve leakage yield. Additionally, the number of CLBs and switches on near critical paths may decrease with larger *k* and *N*, improving timing yield. Wong et al. studied the impact of *N* and *k* on timing and leakage yield [16]. They observe that while *N* has little impact on yield, $k = 7$ maximizes timing yield, $k = 4$ maximizes leakage yield, and $k = 5$ maximizes combined yield. These results are not surprising since leakage roughly scales with area and timing yield is directly related to delay; these results largely match the *k* values that optimize delay, area, and area-delay product, respectively.

Wire segmentation can have an impact on timing yield for similar reasons as *N* and *k*. For a fixed distance net, smaller values of L_{seg} increase the number of buffers on the path, increasing mean delay but decreasing variance due to delay averaging (Equation (11.3)). Kumar et al. found that compared to an FPGA with 50% $L_{\mathrm{seg}} = 8$ and 50% $L_{\mathrm{seg}} = 4$ segments in a 45-nm predictive technology with $3\sigma/\mu = 20\%$ variation, using a mix of shorter segments can reduce both mean delay and variance [3]. They find that architectures with between 20 and 40% $L_{\mathrm{seg}} = 2$ segments can improve μ_τ by 7.2–8.8% and σ_τ by 8.7–9.3%.

Work to date has focused on revisiting traditional FPGA architecture parameters with the new concern for variation. The evidence so far suggests that variation does not significantly change the way we would select these parameters. This leaves open exploration of more significant changes that might introduce new parameters or structures into FPGA architecture design.

11.3.3 Transistor Sizing

Lower level, device, and circuit parameters can also be optimized to mitigate the impact of parameter variation. Transistor sizing is a common strategy used in ASICs to directly reduce the magnitude of variation. Larger transistors have increased numbers of dopants, and, due to law of large numbers effects, the magnitude of dopant variation will reduce. V_{th} variation can be expressed as a function of transistor dimensions as:

$$\sigma_{V_{th}} \propto \frac{1}{\sqrt{WL}} \tag{11.4}$$

Increasing W in logic transistors can increase variation tolerance, but at the cost of area and energy. There has been no published work quantifying the tradeoffs in resizing FPGA logic transistors, which may be a promising avenue of exploration.

Another option is to adjust the W of transistors located in memory, which can reduce the probability of failure of FPGA SRAM bits [8]. SRAM bits can fail in one of four ways: read upsets, write upsets, hold failures, or access time failures. FPGA configuration memory will typically fail with read or hold upsets; writes only happen at configuration time and timing is irrelevant as they are always in the read state. FPGA data memory failures are dependent on application characteristics; Paul et al. [8] study applications where memories were used as logic such that reads dominate writes and hence design a cell that increases read and access time failure tolerances significantly.

Figure 11.2 shows the skewed SRAM cell. Increasing W for the pull-up transistors in the cross coupled inverters decrease read failure probability by increasing the voltage at which a read upset occurs. Decreasing W of the access transistors makes it more difficult for the internal storage node to be flipped, but increases access time failure probability by slowing down reads. Increasing W of the pull-down transistors compensates for this effect. Paul et al. show two orders of magnitude read upset

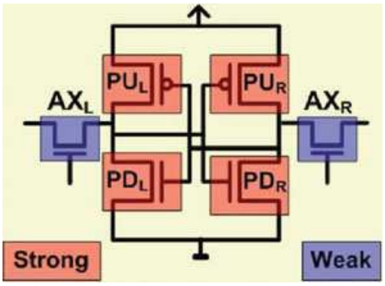


Fig. 11.2
Skewed SRAM cell [8]

and access time failure probability reductions at the cost of a four orders of magnitude increase in write failure probability. The write failures can be addressed with component-specific mapping techniques (Chapter 12).

11.3.4 Asynchronous Design

Choosing the right timing target to achieve high performance and acceptable timing yield is a significant concern with synchronous designs under process variation. If a timing target is chosen too pessimistically, performance of manufactured designs will suffer; if it is chosen too aggressively, many devices may fail. One way to avoid this problem is to design circuits that are not clocked and do not run at a uniform frequency. Instead of using a global clock for synchronization, sequencing can be performed by using handshaking circuitry between individual gates. These self-timed or asynchronous circuits have been studied in depth by ASIC designers and have been explored by FPGA designers as well. Asynchronous FPGAs have been demonstrated in [6, 15] and have even begun to be commercially manufactured [1].

The primary advantage of asynchronous FPGAs in terms of process variation is their high tolerance to delay variability. In the most robust class of asynchronous circuits, quasi-delay insensitive (QDI) circuits [7], only a single timing assumption must be validated to ensure correct operation. This assumption is called an isochronic fork, shown in Fig. 11.3. If the adversary path is faster than the primary path due to delay variation, circuit B may receive an incorrect value. To ensure correctness delay elements can be inserted in the adversary path to margin against error.

Figure 11.4 depicts an asynchronous FPGA LUT circuit from [6]. There are two important characteristics of how asynchronous QDI logic is commonly implemented. First, dual-rail precharge logic is used for high speed circuits. Second, data communication between circuit blocks is done via one-hot encoded signals. We see that in the LUT, precharge signals charge up the dual-rail output bits. During

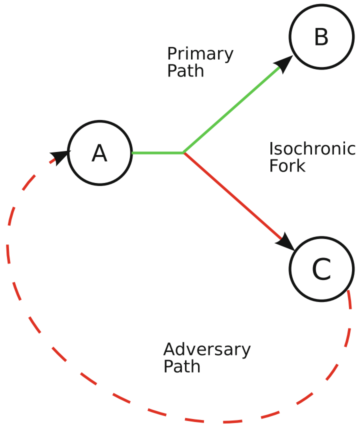


Fig. 11.3 Isochronic fork assumption in QDI asynchronous circuits

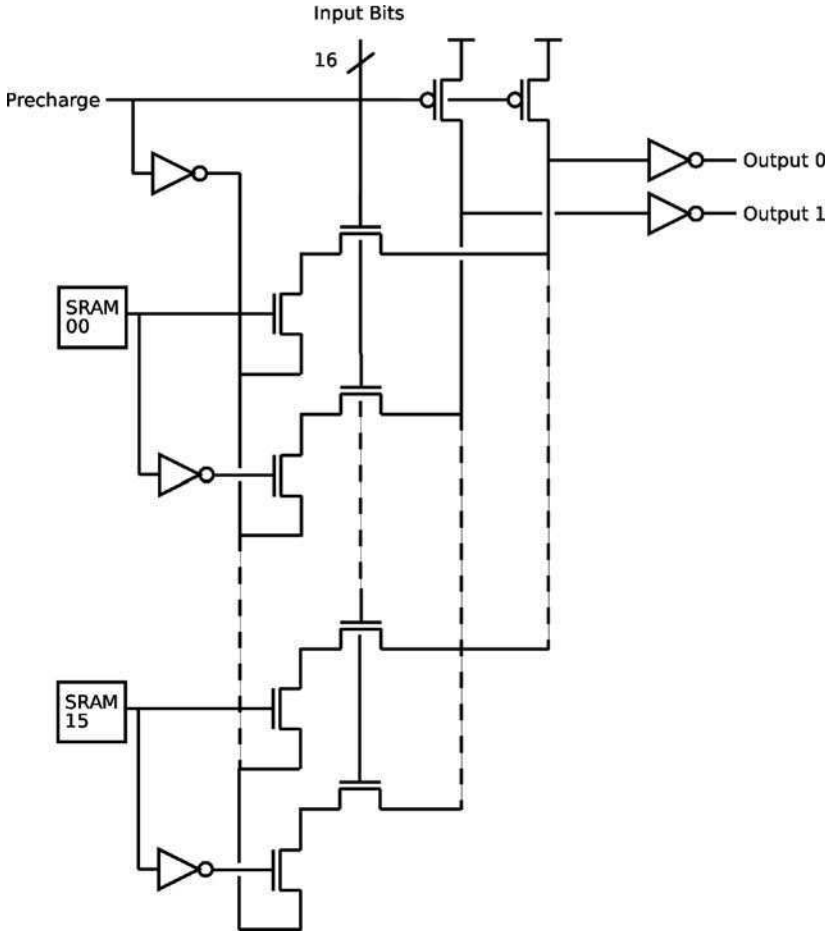


Fig. 11.4 Asynchronous FPGA 4-LUT [6]

evaluate (precharge low), a single LUT input case is selected via the 16-bit, one-hot encoded input, and one of the dual-rail outputs of the LUT is pulled down. The precharge signal is controlled via handshake circuitry from adjacent blocks. When the next block is ready for data and the previous block has sent valid data (as determined by the handshake logic), the LUT will fire.

Figure 11.5 shows a switch circuit. At the heart of the switch circuit is a weak-condition half-buffer, which essentially acts as a register for asynchronous logic. Handshaking signals (enable signals from each direction) control data movement through the switch in the same manner as described for the LUT. Enable and data signals are selected via pass transistor multiplexers as in synchronous switch boxes. One of the primary demonstrated advantages of asynchronous FPGAs is that by using register-like circuits at every switch, interconnect is effectively pipelined for

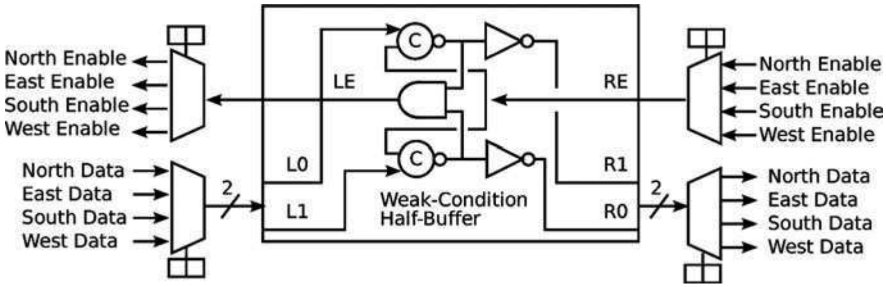


Fig. 11.5 Asynchronous FPGA switch [6]

high speed. Some synchronous FPGA designs have demonstrated highly pipelined interconnect [14], but commercial FPGA vendors have yet to adopt this design style.

If one adopts a fully asynchronous system model, the system will always work regardless of the variability, which reduces the design burden of achieving timing closure in synchronous circuits. The asynchronous design decouples the delay of unrelated blocks allowing each to run at their natural speed. Nonetheless, the throughput and cycle-time of asynchronous circuits are impacted by variation, and high-variation can prevent the asynchronous circuit from meeting a performance yield target as well. The main drawback of asynchronous circuits is that they require area and energy overhead for handshaking circuitry, although energy is simultaneously reduced by eliminating the clock network and through implicit clock gating (as only circuit elements that are performing computation are switching). There has been no published work quantifying the advantages of asynchronous FPGAs under variation, which may be a promising area for future work.

11.4 Impact of Aging

(Chapter 1, Section 1.4) described sources of aging in detail. We will briefly discuss how the primary sources of aging (NBTI, HCI, TDDB, and electromigration) impact FPGAs.

NBTI degrades the threshold of PMOS transistors and is heavily dependent on how long the transistor is held on. The degradation caused by NBTI can be reversed while the PMOS transistor is turned off; this is perhaps the simplest way to combat the induced V_{th} degradation. Reconfiguration potentially allows the FPGA to change the on-profile seen by individual transistors without additional circuitry. NBTI threshold degradation slows down PMOS transistors in logic and significantly reduces the static noise margin (SNM) of SRAM cells. FPGAs typically have more NMOS than PMOS transistors; most architectures use NMOS pass transistors for multiplexers in LUTs and switches that comprise the majority of area. PMOS transistors are used in configuration SRAM, buffers, and embedded structures (e.g., memories and multipliers) in modern parts. Configuration bits are the primary source of NBTI degradation in FPGAs [9] as these transistors are the largest fraction of PMOS devices, and when configured as conducting, they remain on for the entire time the part stays powered on. NBTI can potentially cause severe configuration cell instability through SNM degradation, resulting in early lifetime failures.

HCI degrades the threshold of NMOS devices and is not reversible like NBTI. It is heavily dependent on transistor usage and current density. The most direct technique to combat HCI is to ensure uniform wear leveling across the chip and not use particular NMOS transistors for the entire operational lifetime. Electromigration causes wire faults and is similar to HCI in that it is usage dependent, not reversible, and best avoided through uniform wearing.

TDDB is largely dependent on gate leakage. Hence, the best approach to mitigating TDDB is to reduce leakage current. The prior chapter outlined several techniques to reduce leakage, such as body biasing (Chapter 10, Section 10.4.6), dual V_{th} (Chapter 10, Section 10.4.6), power gating (Chapter 10, Section 10.4.4), and bit inversion (Chapter 10, Section 10.3.3 and Chapter 11, Section 11.5.1).

Finally, each source of aging is heavily dependent on temperature. When compared to CPUs, FPGAs are much more power efficient and have a lower power density. In identical environments for identical workloads, this will translate to lower operational temperatures and longer lifetimes. However, when compared to an equivalent ASIC, FPGAs are much less power efficient, which could mean faster aging.

As in the case of parameter variation, no commercial data about aging has been made public. However, Stott et al. performed accelerated aging experiments on a pair of Altera Cyclone III FPGAs to estimate the impact of aging on FPGAs [13] (Figures 11.6 and 11.7). Using measurement technique from [17], they tested chips overvolted from 1.2 to 2.2 V and heated to 150°C. They observed a frequency degradation of 15%, with NBTI being the primary cause.

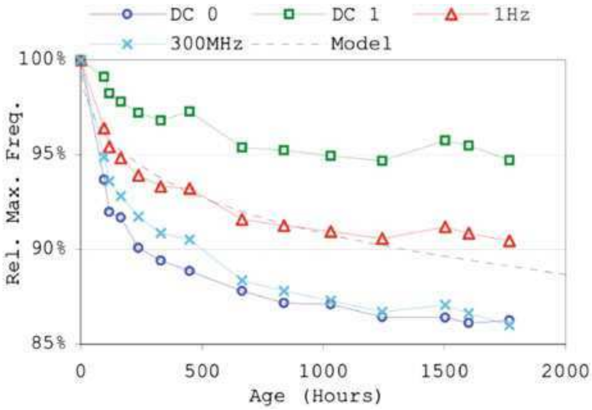


Fig. 11.6 Measured LUT frequency degradation for Altera Cyclone III under accelerated aging [13]

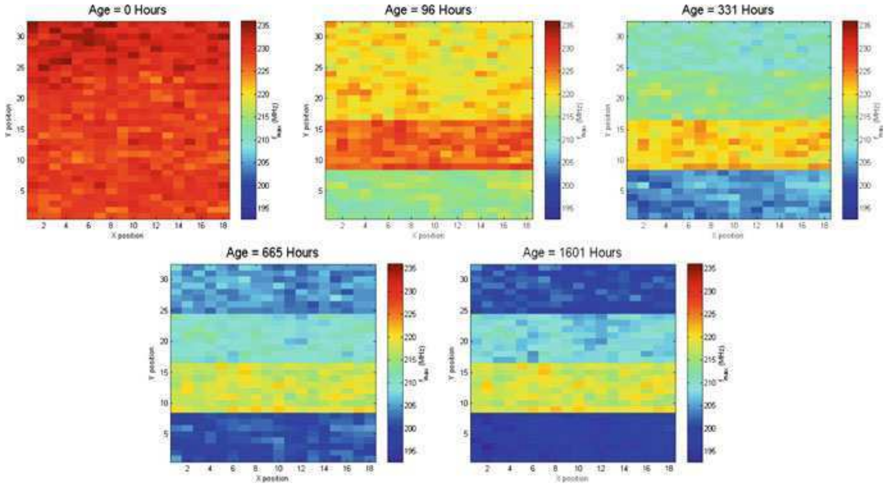


Fig. 11.7 Measured LUT frequency map of Altera Cyclone III under accelerated aging [13]

11.5 FPGA Aging Tolerance

FPGAs are an excellent substrate for compensating for lifetime degradation because of their spare resources and in-field reconfigurability. These characteristics make it easy to change a mapped FPGA design over time to more uniformly spread out wear over the part. The following techniques begin to take advantage of this configurability.

11.5.1 Bit Inversion

Bit inversion was discussed in the previous chapter (Chapter 10, Section 10.3.3) as a way to lower leakage power in LUTs. Because certain LUT configurations leak more than others, by flipping LUT configuration bits (and inverting the input select signals to the LUT to ensure correct output), leakage can be reduced. The same technique can be applied to reducing lifetime degradation in LUTs from NBTI. Because NBTI is reversible, by turning off PMOS transistors that were previously on, degraded values of V_{th} can begin to return to normal.

Ramakrishnan et al. proposed a scheme for relaxing PMOS LUT transistors by loading in a new, bit-inverting bitstream during the life of an FPGA [9]. Inverting configuration bits in the interconnect is more challenging because routes between CLBs may be disrupted. To address this challenge they classified switch configuration bits in three ways: dead, inactive, and used. Dead switches have undriven inputs and unused outputs and can safely be flipped without affecting any routes.

Inactive switches have driven inputs and unused outputs, meaning that inversion could cause a short. They propose a method for flipping these bits in a round-robin fashion to avoid creating a short. Finally, used switches have driven inputs and used outputs. The switches contain valid routes that must be rerouted to allow for bit flipping. They propose a rerouting scheme described in [12]. With these techniques they report a failure-in-time (FIT) decrease of 2.5%.

Bit inversion was also used as a compensation strategy for TDDB [12] by the virtue of it reducing leakage current. Using simulations, they found that bit inversion improved mean-time-to-failure (MTTF) due to TDDB by 24% on average.

11.5.2 Reconfiguration

The most promising technique to use for FPGA lifetime extension is reconfiguration. As an FPGA ages, new configurations can be mapped to a part, avoiding aged resources and utilizing fresh ones or simply rearranging the assigment of logic to resources. Srinivasan et al. explored using reconfiguration to compensate for both HCI and electromigration [12]. Both effects are irreversible and usage dependent, and require avoidance of over-utilization of resources over time.

To avoid HCI, Srinivasan et al. propose to re-place a design over time to previously unused portions of the FPGA. Figure 11.8 shows an example. Using switching activity estimates, they determine the extent to which a region has aged during its operational lifetime. They then perform re-placement using an algorithm that can place blocks with regional restrictions, avoiding areas estimated to be degraded by HCI. Because placement is regionally constrained potentially in a smaller space, the new placement may have a longer critical path. They report that using this method to avoid HCI, part life can be increased by 28% at the cost of 1.53% delay.

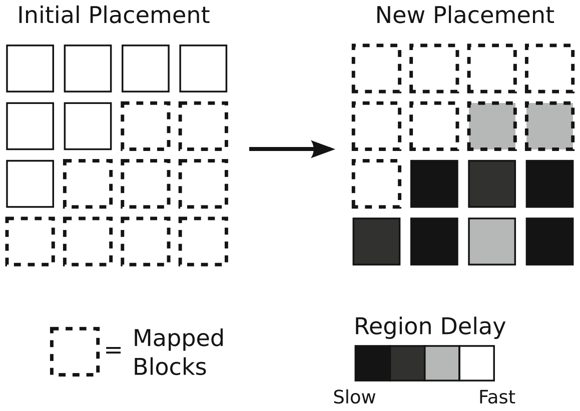


Fig. 11.8 Region relocation for uniform wear

Since electromigration targets wires, simply re-placing logic may not ensure avoidance of specific interconnect segments. Therefore, to mitigate electromigration Srinivasan et al. propose to re-route nets over time. Only nets that have high activity factors and are most likely to fail are chosen for re-routing. Similar to re-placement, re-routing incurs a frequency penalty as re-routed nets may not be timing optimal. By using this technique to combat electromigration they report a lifetime extension of 14% at the cost of 1.3% delay.

An important point to note about these techniques is that they assume that every component is the same and ages in the same way. Hence, every chip will be remapped in the same manner over its lifetime. As manufactured chips are all unique due to process variation and age in unique ways, it would be more effective to create customized configurations for chip. However, this scheme would require ways to measure and characterize each chip separately, in addition to generating a custom mapping per chip. The next chapter will explore these ideas and challenges in depth.

11.6 Summary

Table 11.1 summarizes techniques for variation tolerance; Table 11.2 summarizes techniques for aging tolerance. As in the case of low power techniques, benefits may not be cumulative.

Table 11.1 Roundup of FPGA techniques for variation tolerance

	$3\sigma/\mu$ V_{th} variation		Timing improvement		Yield improvement	
Technique	Regional	Random	μ	σ	Timing	Leakage
SSTA clustering	10%	6%	5.0%	6.4%	9.9%	–
SSTA placement	10%	6%	4.0%	6.1%	8.0%	–
SSTA routing	10%	6%	1.4%	0.7%	2.2%	–
Logic block architecture	?	?	–	–	9%	73%
Interconnect architecture	–	20%	7.2–8.8%	8.7–9.3%	–	–

Table 11.2 Roundup of FPGA techniques for aging tolerance

	Target			
Technique	Aging mechanism	Location	Lifetime metric	Lifetime increase (%)
Bit inversion	NBTI	Configuration Bits	FIT	2.5
Bit inversion	TDDB	LUTs	MTTF	24
Re-placement (open loop)	HCI	LUTs	MTTF	28
Re-routing (open loop)	Electromigration	Interconnect	MTTF	14

These techniques show that FPGAs can begin to tolerate variation by using very similar techniques as those used by ASICs. However, most of these techniques do not use the most powerful tool at the disposal of an FPGA: reconfiguration. FPGAs have a valuable opportunity to surpass ASICs in reliability at a time when variation and aging effects are predicted to increase substantially. The fundamental problem caused by variation is that every manufactured chip is degraded differently; component-specific reconfiguration is the natural response to compensate for this unique degradation on a per-chip basis. For aging we saw that open-loop configurability can increase lifetime; however, by closing the loop and measuring characteristics of individual components, reconfiguration should be able to extend life even further. The challenges and potential benefits of component-specific mapping will be addressed in the next chapter.

References

1. Achronix Semiconductor Corporation, Inc. (2010) Achronix Speedster Data Sheet, preliminary (v1.0) edn http://www.achronix.com/
2. Katsuki K, Kotani M, Kobayashi K, Onodera H (2005) A yield and speed enhancement scheme under within-die variations on 90 nm LUT array. In: Proceedings of the IEEE custom integrated circuits conference, pp 601–604
3. Kumar A, Anis M (2010) FPGA design for timing yield under process variations. IEEE Trans VLSI Syst 18(3):423–435
4. Lin Y, He L, Hutton M (2008) Stochastic physical synthesis considering prerouting interconnect uncertainty and process variation for FPGAs. IEEE Trans VLSI Syst 16(2):124
5. Lucas G, Dong C, Chen D (2010) Variation-aware placement for FPGAs with multi-cycle statistical timing analysis. In: Proceedings of the international symposium on field-programmable gate arrays. ACM, New York, NY, pp 177–180
6. Manohar R (2006) Reconfigurable asynchronous logic. In: Proceedings of the IEEE custom integrated circuits conference, pp 13–20
7. Martin A, Nystrom M (2006) Asynchronous techniques for system-on-chip design. Proc IEEE 94(6):1089–1120
8. Paul S, Mukhopadhyay S, Bhunia S (2009) A variation-aware preferential design approach for memory based reconfigurable computing. In: Proceedings of the international conference on computer-aided design. ACM, New York, NY, pp 180–183
9. Ramakrishnan K, Suresh S, Vijaykrishnan N, Irwin M (2007) Impact of NBTI on FPGAs. In: Proceedings of the international conference on VLSI design. IEEE Computer Society, pp 717–722
10. Sedcole P, Cheung PYK (2006) Within-die delay variability in 90 nm FPGAs and beyond. In: Proceedings of the international conference on field-programmable technology, pp 97–104
11. Sivaswamy S, Bazargan K (2008) Statistical analysis and process variation-aware routing and skew assignment for FPGAs. Trans Reconfig Technol Syst 1(1):1–35. doi: http://doi.acm.org/10.1145/1331897.1331900
12. Srinivasan S, Mangalagiri P, Xie Y, Vijaykrishnan N, Sarpatwari K (2006) FLAW: FPGA lifetime awareness. In: Proceedings of the ACM/IEEE design automation conference. ACM, p 635
13. Stott EA, Wong JSJ, Pete Sedcole P, Cheung PYK (2010) Degradation in FPGAs: measurement and modelling. In: Proceedings of the international symposium on field-programmable gate arrays. ACM Press, New York, NY, p 229

14. Tsu W, Macy K, Joshi A, Huang R, Walker N, Tung T, Rowhani O, George V, Wawrzynek J, DeHon A (1999) HSRA: high-speed, hierarchical synchronous reconfigurable array. In: Proceedings of the international symposium on field-programmable gate arrays, pp 125–134
15. Wong C, Martin A, Thomas P (2003) An architecture for asynchronous FPGAs. In: Proceedings of the international conference on field-programmable technology, pp 170–177
16. Wong H, Cheng L, Lin Y, He L (2005) FPGA device and architecture evaluation considering process variations. In: Proceedings of the international conference on computer-aided design. IEEE Computer Society, p. 24
17. Wong JSJ, Sedcole P, Cheung PYK (2008) A transition probability based delay measurement method for arbitrary circuits on FPGAs. In: Proceedings of the international conference on field-programmable technology, pp 105–112
18. Wong JSJ, Sedcole P, Cheung PYK (2009) Self-measurement of combinatorial circuit delays in FPGAs. Trans Reconfig Technol Syst 2(2):1–22

Chapter 12
Component-Specific Mapping for Low-Power Operation in the Presence of Variation and Aging

Benjamin Gojman, Nikil Mehta, Raphael Rubin, and André DeHon

Abstract Traditional solutions to variation and aging cost energy. Adding static margins to tolerate high device variance and potential device degradation prevent aggressive voltage scaling to reduce energy. Post-fabrication configuration, as we have in FPGAs, provides an opportunity to avoid the high costs of static margins. Rather than assuming worst-case device characteristics, we can deploy devices based on their fabricated or aged characteristics. This allows us to place the high-speed/leaky devices as needed on critical paths and slower/less-leaky devices on non-critical paths. As a result, it becomes possible to meet system timing requirements at lower voltages than conservative margins. To exploit this post-fabrication configurability, we must customize the assignment of logical functions to resources based on the resource characteristics of a particular component after it has been fabricated and the resource characteristics have been determined—that is, *component-specific mapping*. When we perform this component-specific mapping, we can accommodate extremely high defect rates (e.g., 10%), high variation (e.g., $\sigma_{V_t} = 38\%$), as well as lifetime aging effects with low overhead. As the magnitude of aging effects increase, the mapping of functions to resources becomes an adaptive process that is continually refined in-system, throughout the lifetime of the component.

12.1 Introduction

A key difference between FPGA s and ASICs is that functions are assigned to hardware resources *after* the FPGA chip has been fabricated. This gives us a unique opportunity with FPGAs—we can adapt the assignment of functions to resources knowing which transistors are functional, fast, and leaky. That is, if we customize

B. Gojman (✉)
Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, USA
e-mail: bgojman@seas.upenn.edu

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1_12,

the assignment on a per-component basis, we can avoid the bad resources and select where to use fast or low power resources in our design. Since we can choose not to use a resource, we can avoid being driven by worst-case resource characteristics and the uncommon tails of the device characteristics distribution. As we will see, in some cases, this choice allows us to use a resource beneficially when it would otherwise have been unusable (Section 12.6.3). This capability may be essential for extremely scaled or bottom-up manufactured devices (e.g., nanowire-based PLAs in Section 12.5). This component-specific mapping allows us to avoid defective and unusable devices at little additional cost over the base FPGA, operate faster or at lower voltages than a statistical design by avoiding slow or high threshold voltage devices (Sections 12.6.1 and 12.6.2), and repair degradation due to aging throughout a component's lifetime (Section 12.8).

Nonetheless, component-specific mapping breaks our traditional one-mapping-fits-all model for FPGAs. It requires us to abandon the increasingly false abstraction that all FPGA chips are identical. As a result, we must generate a potentially unique mapping for every component. Figure 12.1 shows the steps involved in component-specific mapping and contrasts this with the traditional one-mapping-fits-all model.

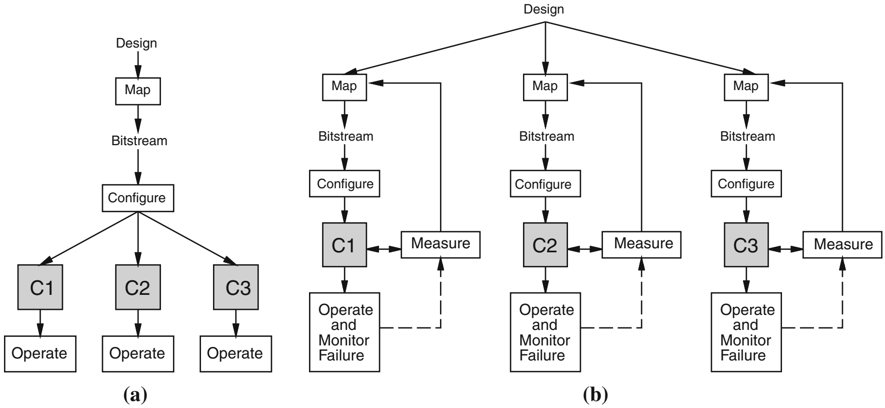


Fig. 12.1 Steps in component-specific mapping contrasted with the conventional one-mapping-fits-all model (**a**) One-mapping-fits-all (**b**) component-specific mapping

Component-specific mapping introduces many new challenges for FPGA use.

- How do we measure the resource characteristics and capabilities? (Section 12.3)
- How completely and precisely do we need to know device characteristics? (Sections 12.6.4 and 12.7.2)
- How do we keep the unique mapping time low? (Section 12.7.1)
- How do we fit this mapping into our FPGA use model without unduly complicating handling and assembly? (Section 12.7.1)
- How do we detect changes in device characteristics and adapt to them? (Section 12.8.3).

Component-specific mapping effectively opens up new dimensions in our design space. For example, it gives us the opportunity to trade some additional measurement and mapping time for increased yield, decreased energy, and increased performance. It also opens up tradeoffs in FPGA configuration bitstream size and load time. These new tradeoffs give us an opportunity to shift costs from critical and expensive dimensions such as energy into dimensions where we can better tolerate additional costs such as burnin and load time.

In this chapter we outline the prospects for component-specific mapping and highlight some of the current research in this promising new area. We start with the basic idea (Section 12.2), then highlight measurement techniques (Section 12.3) and some of the simplest manifestations of the idea (Section 12.4). We review the NanoPLA in Section 12.5 as background for later sections that use NanoPLA challenges and solutions to illustrate how component-specific mapping accommodates variation and defects. Section 12.6 explores mapping to accommodate and exploit variation when we have full knowledge of device characteristics. Section 12.7 looks at defect-tolerance approaches, including ones that operate with little *a priori* information on the resource characteristics. Section 12.8 highlights opportunities for in-system repair and reviews approaches for detection of in-system failures, and Section 12.9 wraps up this chapter with speculations on the future systems. As this is a young area of exploration, many questions remain open and many aspects merit further research. This chapter provides an introduction and guide to the key issues and opportunities.

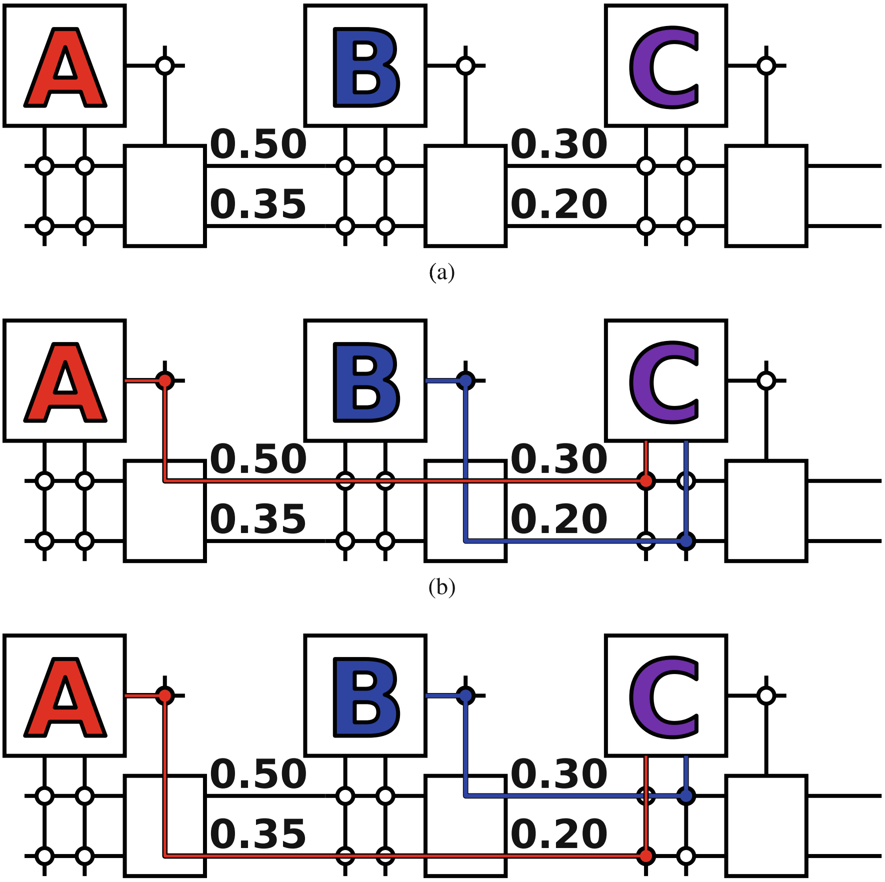
12.2 Motivating Example

To illustrate the basic idea and potential benefits of post-fabrication, knowledge-based mapping, we start with a simple, illustrative example. The goal will be to reduce the overall power consumed by carefully selecting which routes use which resources based on the component characteristics.

Figure 12.2a shows a cartoon version of simplified FPGA organization with three blocks and two channels buffered at every switch. The three labels on the blocks show which function is produced by each. The four labels on the four channel segments indicate the threshold voltage of the switch driver for each segment in volts. The expected nominal threshold voltage $V_{\mathrm{th}_{\mathrm{Nom}}}$ is 0.30 V. However, due to variation, only one out of the four segments has its nominal value.

Assume that our goal is to compute $C = A + B$ within 125 ps. After placement, function A is on the first block and B on the second. It is the third block's responsibility to compute $A+B$ to produce C. Assume C has a gate delay of 25 ps. Therefore, to meet our timing target the interconnect must take at most 100 ps. Figure 12.3 shows the effect that the source voltage V_{dd} has on the delay of a segment for multiple threshold voltages V_{th}. As the threshold voltage increases, it is clear that a higher V_{dd} is required to meet the same timing.

Oblivious to the variation present in the FPGA, a router may produce the results shown in Fig. 12.2b. The critical path on this circuit is the connection from A to C.



(a)

(b)

(c)

Fig. 12.2 Simple motivational example. (**a**) Simple FPGA showing threshold voltage of the switch driver for each segment; (**b**) Variation oblivious route for $A + B = C$; (**c**) Variation aware full knowledge route for $A + B = C$

Using the graphs in Fig. 12.3 we see that to meet our timing requirement of 125 ps, we must set our supply voltage, V_{dd}, to 0.49 V. This gives $A \rightarrow C$ a path delay of 100 ps and 3 ps for path $B \rightarrow C$. Combined with the 25 ps gate delay yields the required 125 ps.

If, instead, we give a delay-oriented router the fabricated characteristics of each segment, it can find the route shown in Fig. 12.2c. This route avoids the extremely high threshold voltage driver, allowing the supply voltage to be set to $V_{\text{dd}} = 0.32V$ while still meeting our timing requirement. Here, path $A \rightarrow C$ is still the slowest path. Using the full knowledge of component characteristics, the router assigns this path to low V_{th} segments. Since path $B \rightarrow C$ is less critical, the router leverages the available slack to route it through a higher V_{th} segment than in the oblivious route while still meeting timing. At $V_{\text{dd}} = 0.32V$, path $A \rightarrow C$ has a delay of 100 ps.

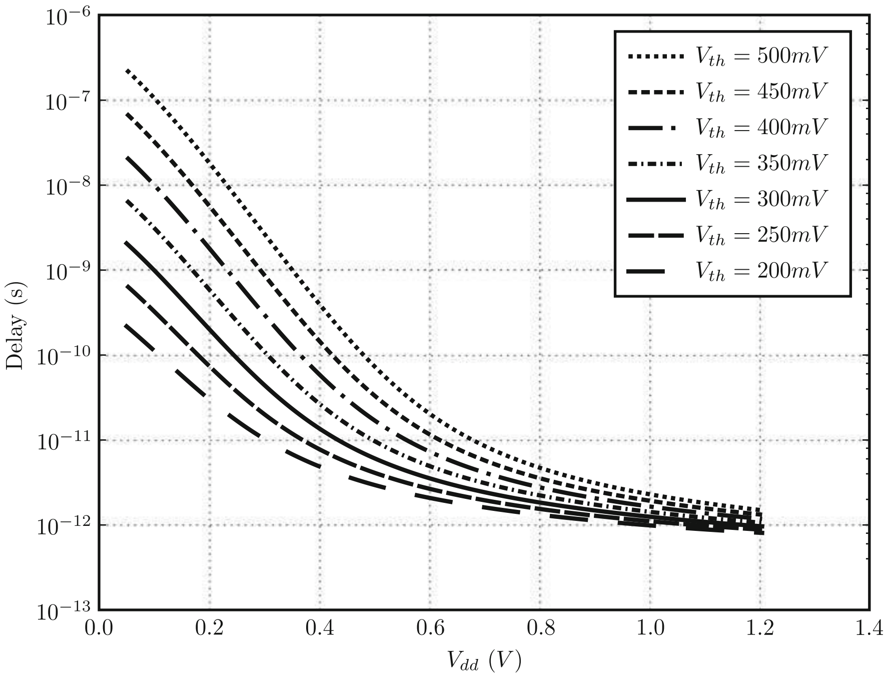


Fig. 12.3 Delay as a function of V_{dd} for an FPGA switch driver at multiple V_{th}s

The delay of path $B \rightarrow C$ is 3.6 ps, it is slower than in the oblivious case, but faster than the parallel $A \rightarrow C$ input so does not impact overall cycle time. In this simple example, using knowledge of the underlying characteristics of the FPGA can reduce the active energy consumption by a factor of $\left(\frac{0.49}{0.32}\right)^2 = 2.3$ compared to the oblivious route.

While this simple case illustrates the use of routing to navigate interconnect resources and reduce power consumption, it should be clear that the general idea can be applied much more broadly. We can perform similar substitution or assignment for all resources (e.g., logic, memory). This strategy can help us mitigate the effects of defects, variation, and aging. Mapping based on current characteristics of resources can be exploited in placement, memory assignment, clustering, and function binding in addition to routing. As we will see, this technique gives us the opportunity to accommodate very high defect and variation rates with low overhead beyond the base FPGA costs.

12.3 Measurement

As the example above illustrates, component-specific mapping demands that we identify the post-fabrication characteristics of the FPGA. Since time on expensive test equipment can already be a significant fraction of integrated circuit production costs, it is reasonable to be concerned that the required cost of measurement might be prohibitive. Furthermore, the characterization required here is more fine-grained than the normal pass/fail testing for chips, potentially requiring even larger test time than conventional circuits.

In this section, we review approaches to measuring delays of resources on FPGAs without requiring expensive test equipment (See Table 12.1). Section 12.3.1 describes the use of scannable registers to test and measure resource characteristics, while Section 12.3.2 describes how to configure testing apparatus on the FPGA itself.

Table 12.1 On-chip measurement options

Technique	Sec.	Advantage	Disadvantage
DC test scan	12.3.1.2	minimal requirements; minimally intrusive	DC defects only; low throughput
At-speed scan testing	12.3.1.3	inexpensive; minimally intrusive	low throughput; no jitter, self-heat
Configured ring oscillator	12.3.2.1	high speed	requires reconfiguration; displaces logic
Configure path test	12.3.2.2	high speed; arbitrary resources and paths	requires reconfiguration; displaces logic

12.3.1 Scan-Based Timing Measurement

Modest register scan support can be used to measure the register-to-register transition delay of synchronous paths. The basic idea is that we can (a) load an initial value into a set of registers, (b) load a value to which the register should transition, (c) allow the full-speed system clock to run for two steps, clocking the transition into the source registers and then clocking the result into the downstream registers, and (d) scan out the value captured at the output registers to see if it represents the correct result of the transition. Scan support is relatively inexpensive. The serial access to load and readback the registers keeps overhead for controllability and observability low but also makes test throughput low. Nonetheless, scan support has been heavily used for measuring the delay of timing paths in ASICs and can be similarly be used to measure component-specific delays in FPGAs. Most of the scan access support is already present in some form in typical FPGA designs.

12.3.1.1 Scannable Registers

Figure 12.4 shows a register with scan support. The *scan_in* to *scan_out* signals allow the scan flip-flops to be connected as a long, serial shift register. In scan shift mode, *scan_clkA* and *scan_clkB* serve as a two-phase clock and shift data through the scan registers, allowing us to serially shift data into and out of the scan registers.

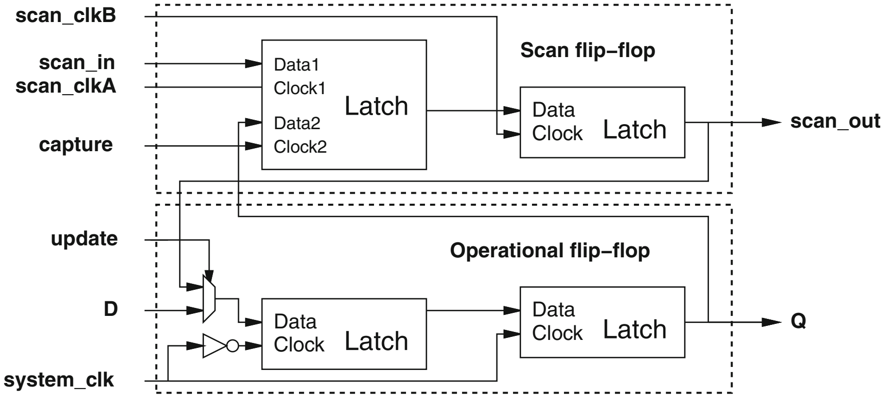


Fig. 12.4 Scannable register

The *capture* and *update* signals allow the scan register to observe or control the logic design. When *capture* is asserted, it loads data from the operational flip-flop into the scan flip-flop's slave latch instead of from the shift register. This effectively performs a parallel load into the scan shift register from the operational flip-flops across the entire chip. After performing this capture operation, we can return the scan register to shift mode in order to read the value of the operational flip-flops out of the chip. Alternately, when *update* is asserted, the operational flip-flop loads from the master scan latch rather than from its normal input. In this way, we perform a parallel load from the scan shift register to overwrite values in operational registers.

For typical operation, a Test Access Port (TAP) controller determines which paths are active, possibly disabling the system clock while in various scan operational modes. IEEE1149.1 is a standard definition for a 4-pin TAP interface at the board level [21]. Figure 12.5 shows typical architecture for the scan control. Xilinx FPGAs use a similar technique to the scannable registers to set and capture register values through the configuration path [96, 97].

12.3.1.2 DC testing

Using this structure, we can perform DC testing for stuck-at failures, shorts, and opens. Assuming that all paths begin and end from scannable registers, we can shift in a pattern, perform an *update*, allow the system clock to fire once, *capture* the value currently seen by the operational registers, and scan the values out. This allows us to apply test vectors and observe the circuit and interconnect response. A properly chosen set of test vectors can verify that all interconnect paths properly transfer both 0's and 1's from the intended sources to destinations, that logic functions perform their intended operations, and that no adjacent paths interfere with each other. Boundary-scan connectivity tests were originally designed with these scan registers on the IO pins of chips in order to test connectivity among chips at the board level

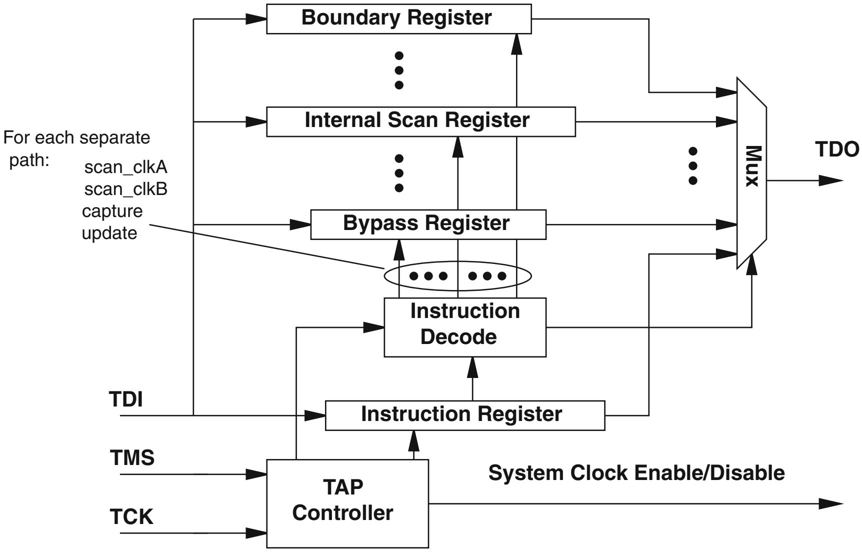


Fig. 12.5 Test Access Port (TAP) architecture for scan control

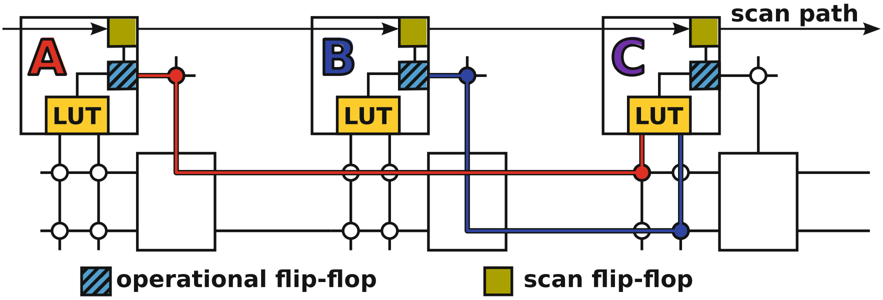


Fig. 12.6 Scan test example

[65]. By making the registers inside the chip scannable as well, we allow similar testing for the logic and interconnect within the chip.

Figure 12.6 shows the simplified FPGA architecture from Section 12.2 with scan registers. To test the function C=op(A,B), we can use the scan path to set the register outputs of blocks A and B, perform an *update* then a *capture*, and then use the scan path to read out the resulting value of C. This tests both the proper function of the logic and the proper connectivity on the interconnect between the blocks.

12.3.1.3 At-Speed Testing

For modern chips, we are concerned both with DC failures, such as shorts and opens, and delay failures due to high variation of parameters during manufacturing. With standard, low-speed, serial scan operation, the time between the *update* of source register and the *capture* at the sink register can be multiple cycles of the scan clock. Since the scan clock is slow compared to the system clock, the time between *update* and *capture* can be very long compared to the intended clock frequency for the chip. Consequently, these tests do not tell us how fast we can clock the path.

It is possible to perform at-speed tests by carefully generating a pair of pulses from the system clock at the intended operational speed [57, 68, 75]. After we have scanned a test value into the shift register, it does not become visible to the operational registers until we clock the system clock after an *update*. If we can arrange to pulse the system clock once, it is clocked into the system register synchronous to the system clock. Further, if we can arrange for a second pulse of the system clock at its operational period, we can capture the result of this transition after one system clock period at all downstream registers. For the design shown in Fig. 12.4, proper timing requires that we disable the *update* signal synchronous with the first system clock pulse so the second clock pulse allows the operational flip-flop to load data from its normal input. If the values can propagate properly through the logic and interconnect within one clock period, the downstream registers will store the correct value. If they cannot, the downstream registers will store incorrect values. After these two at-speed clock pulses, we can use the scan capture mode to sample the values on the operational registers and scan them out as before. The additional support to handle this timing test is a scan controllable test operation that can trigger a pair of full-speed system clock pulses following an *update*. Figure 12.7 shows sample signal timing for this at-speed test mode.

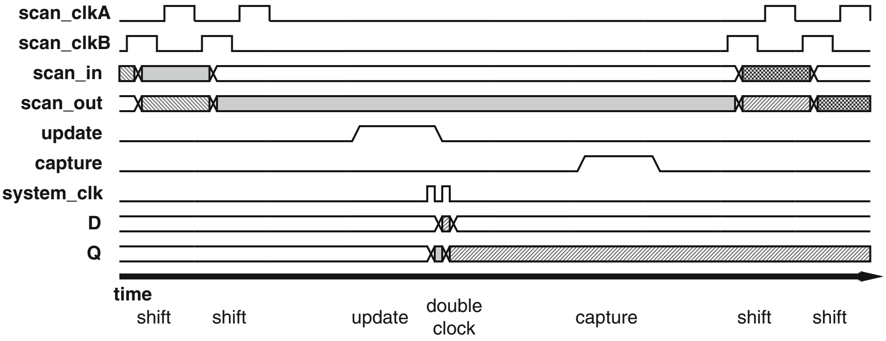


Fig. 12.7 Timing for at-speed testing

Returning to our example in Fig. 12.6, we can test signal propagation using this at-speed test. For illustration, assume that C is implementing an OR function and we want to test if a low to high transition on A can reach C within an operational clock cycle. We first scan a 0 into A and a 0 into B and perform an *update*. This sets A and B to driving 0's and leaves C computing a 0 result. We then scan a 1 into A and a 0 into B and perform the at-speed *update* and double-clock operation. The first operational clock forces the operational flip-flop at A to transition from a 0 to 1, beginning its propagation through the interconnect. The second operational clock allows C's operational flip-flop to capture the value it sees at the clock edge. We can then perform a *capture* operation and scan out the register values in order to see if A's transition from 0 to 1 propagated fast enough through the interconnect and LUT to change C's register value to 1.

If we can further adjust the frequency of the system clock, we can use this technique to determine the speed of a path. If we succeed in capturing the correct values at a given clock frequency, we know that the circuit delay is less than that test clock period. If we only capture the old data, we know it is higher. Using these binary indications, we can adjust the frequency to identify the highest frequency at which the entire circuit, or maybe just a particular path, can operate correctly.

12.3.1.4 Strengths and Weaknesses

While a scannable register is larger than a non-scan register, the register area is small compared to the LUT and interconnect in an FPGA. As such, scan support for all registers in an FPGA is relatively inexpensive. The tests can be driven from an inexpensive, external scan-based tester or an inexpensive on-board tester. These tests can be performed without requiring reconfiguration of the logic, testing end-to-end operation when the component is configured. This arrangement allows the scan test to also verify proper configuration and the proper operation of configuration cells. Since data is serially shifted in and out of the scan path using a low-speed clock, testing throughput is low. The need to adjust the system clock frequency and retest at different frequencies further increases system test time. Since high-speed tests only involve a couple of high-speed clock pulses surrounded by relatively long idle periods, the chip is not switching at its full rate between tests. This means that the chip will not see the full temperature impact of self-heating during these scan-based timing tests.

12.3.2 Configuring Self-Measurement Designs

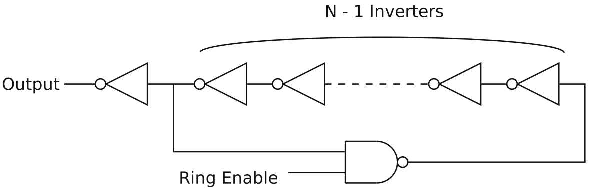
We can exploit the fact that FPGAs can be reconfigured to perform different functions to build test and measurement structures into the FPGA. This allows us to measure fine-grained resource sets without additional overhead. We can configure rich testing structures for measurement periods, and then remove them during actual operation. Configured test structures can be highly parallel, increasing the throughput of test compared to scan-based designs.

12.3.2.1 Ring Oscillator Measurements

Variation in semiconductor processes is frequently characterized via test structures. Different circuits and measurement techniques can be used for measuring both regional, spatially correlated variation and uncorrelated, random variation. A standard test structure for measuring regional variation is an array of ring oscillator circuits [12] (also see Chapter 1, Section 1.5.4). By measuring the frequency of each oscillator one can determine if the region local to the ring oscillator is slow (high V_{th}) or fast (low V_{th}). As individual transistor delays or V_{th}'s are not being measured, uncorrelated, random variation cannot be characterized using a single ring oscillator. Additionally, the actual variability of fabricated ASIC designs can only be estimated and bounded from test chip measurements as only test chips are being measured. FPGAs, however, can use the same substrate for both measurement and computation by configuring oscillators for measurement and later configuring a real design. Sedcole et al, performed the first delay variability characterization of an entire commercial FPGA with this technique [76].

N-stage ring oscillators (Fig. 12.8) can be configured on the FPGA, where each stage delay is the delay through a single LUT, interconnect switches, and associated wiring. Connecting the output of the ring oscillator to a counter allows us to count the number of oscillations within a known clock period and compute the ring oscillator frequency. A NAND gate can be used to enable/disable the oscillator so the count is taken for a well-defined period of time. A ring oscillator only requires at minimum N=3 stages. However, at N=3, the oscillator frequency may be too high for reliable operation of the counter. In practice values of N=5 or N=7 are required. This limits the granularity of measurement to five stages of LUT + interconnect delays. Although individual stage delays cannot be obtained, individual stage delay variance can be estimated by dividing the oscillator delay variance by the number of stages. Three separate estimates can be made and correlated by measuring the delay of the N=5 ring, the N=7 ring, and the difference between the two measurements:

Fig. 12.8 Ring oscillator



$$\sigma^2_{\text{stage}} \approx \frac{\sigma^2_{N=7}}{7} \approx \frac{\sigma^2_{N=5}}{5} \approx \frac{\sigma^2_{\text{difference}}}{3} \tag{12.1}$$

Because this measurement technique lumps together LUT and interconnect delays, it is more useful for characterizing and isolating LUT delay variability than for characterizing the delay of specific resources. The impact of interconnect delay on ring-oscillator frequency can be minimized by measuring LUTs within the same CLB, utilizing local interconnect rather than higher level routing.

Figure 12.9 shows a possible chip-level measurement scheme. Ring oscillators are configured in an array on each CLB within the device. If we run a single ring oscillator at a time, we can share the counter across oscillators. Selection circuitry controls row and column decoders to enable individual oscillators. A timer ensures that an oscillator is only enabled for a short period of time to avoid local self-heating effects. A multiplexer selects which ring oscillator is allowed to clock the counter.

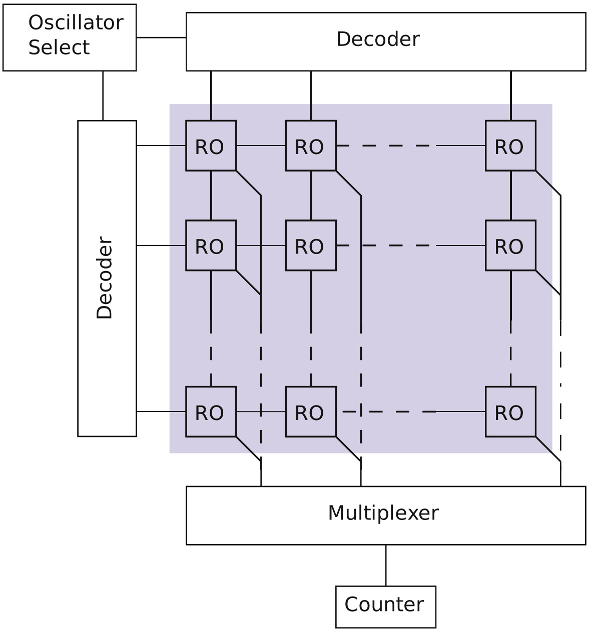


Fig. 12.9 Measurement array of ring oscillators and associated support circuitry [76]

12.3.2.2 At-Speed Path Delay Measurements

Ring oscillators are useful for measuring FPGA delay variability but are not fine-grained enough to measure the delay of an individual resource and are difficult to use for higher levels of interconnect. Additionally, as they do not represent real circuits mapped to FPGAs, actual oscillator delay measurements can only be used indirectly for component-specific mapping.

Section 12.3.1.3 described how at-speed testing can be used to determine how fast an FPGA design can be clocked using scan register support. Instead of using scan registers, specific delay testing structures can be configured in the FPGA fabric itself to assess how fast individual paths can be clocked. With this method the delay of many paths can be measured in parallel and in rapid succession, without pausing to scan out the entire contents of the FPGA's registers.

Figure 12.10 shows the configured measurement circuit for testing a single combinational path which can consist of any number of LUT and interconnect delays [93]. A test stimulus is clocked into the launch register, through the combinational path, and into a sample register. The clock rate is generated by a test clock generator circuit. Conveniently, modern commercial FPGAs contain clock generators with picosecond timing precision. Increasing clock rates are applied for each test, and the input and output of the sample register are compared to determine if the correct value was latched for the tested clock frequency. If an error is detected a status register is set, indicating a timing failure.

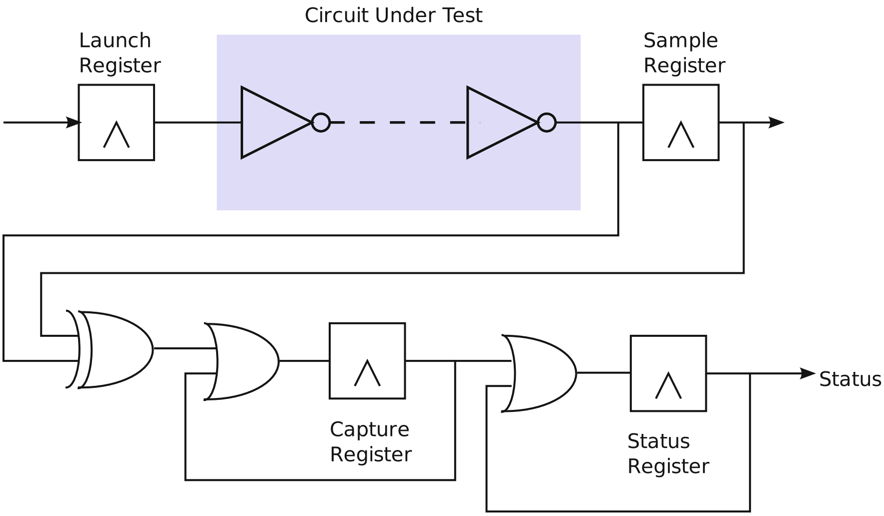


Fig. 12.10 Path delay measurement circuit [93]

With this strategy, precise combinational delays of real circuits can be obtained solely through configuration and self-measurement. Only three cycles are required per test, and one test can be performed immediately after another (without the scan chain delays in Fig. 12.7). Further, testing can be applied in parallel, where status register outputs can be combined into words and written in parallel into on-chip FPGA memory. Wong et al. [93] report full self-characterization of all LUTs (using internal CLB paths, as in the ring oscillator case) on a commercial FPGA in 3 s. Arbitrary path delay measurements using custom-embedded structures such as carry-chains and embedded multipliers were also demonstrated for commercial parts.

12.3.2.3 Resource Isolation Measurements

While it may not be possible to directly measure the delay of individual primitive resources (e.g., LUT delays, interconnect segment delays) in an FPGA using a ring oscillator or clocked path test, that does not prevent us from discovering the delay of individual elements. In particular, since we have the ability to reconfigure the device, we can configure different rings or chains of resources, make multiple measurements, and calculate the necessary delay contribution of each constituent.

For the sake of illustration, consider the task of estimating the delay of individual LUTs in a CLB. Section 12.3.2.1 described how we can measure the frequency of a five-stage LUT ring oscillator. If there are more than five LUTs in each CLB (as in Stratix Logic Array Blocks [54, 55]), we can measure oscillators with different subsets of the LUTs in order to estimate the delay of each LUT. That is, each measurement gives us an equation like:

$$2\,(D_1 + D_2 + D_3 + D_4 + D_5) = 1/F_6 \tag{12.2}$$

Where D_i is the delay of LUT i and F_j is the frequency estimated for the oscillator omitting LUT j. As long as we have m linearly independent equations in m unkowns, we can solve for the m unkowns. So, for example, if we collect:

$$\begin{array}{rcl}
D_1 + D_2 + D_3 + D_4 + D_5 \phantom{{}+D_6} & = & 1/(2F_6) \\
D_1 + D_2 + D_3 + D_4 + \phantom{D_5 + {}} D_6 & = & 1/(2F_5) \\
D_1 + D_2 + D_3 + \phantom{D_4 + {}} D_5 + D_6 & = & 1/(2F_4) \\
D_1 + D_2 + \phantom{D_3 + {}} D_4 + D_5 + D_6 & = & 1/(2F_3) \\
D_1 + \phantom{D_2 + {}} D_3 + D_4 + D_5 + D_6 & = & 1/(2F_2) \\
D_2 + D_3 + D_4 + D_5 + D_6 & = & 1/(2F_1)
\end{array}$$

We can solve for each D_i. For example:

$$D_1 = \frac{1}{10}\left(\frac{1}{F_2} + \frac{1}{F_3} + \frac{1}{F_4} + \frac{1}{F_5} + \frac{1}{F_6} - \frac{4}{F_1}\right) \tag{12.3}$$

Full elaboration and estimate of resource delays for a real architecture is much more complicated. Nonetheless, this same basic idea can be extended to determine the delay of other individual resources such as segments and CLB inputs. It will take many reconfigurations to collect all the data required, but many devices can be measured in each configuration using techniques similar to the array of ring oscillators above. The number of configurations needed does not scale with the size of the array.

12.3.2.4 Strengths and Weaknesses

Configuring self-measurement designs enable high throughput measurements as many paths can be tested in parallel. By avoiding scan chains to write test inputs and read results, these designs can apply test patterns at much higher speeds. Higher speed tests can enable larger scale measurements that may need millions of cycles of data (e.g., for skew, jitter, self-heating) that are not practical with scan-based measurements. Additionally, because these techniques are configured on the FPGA fabric rather than embedded in the fabric, test structures with much more complexity than a simple scan register are possible. The drawbacks of self-measuring designs are that they require resources to map the test structures, which can displace user logic. Hence, several configurations may be necessary to measure resources that may be occupied by the configured test structures. These reconfigurations can be time consuming. Because the test structures are being mapped to resources that need to, in fact, be tested, test structures are subject to the same variations and defects as all resources.

12.4 Early Ideas

Fully exploiting the potential for component-specific mapping requires a resource measurement strategy and mapping tools that can accommodate the fabricated characteristics of individual parts. Nonetheless, a few approaches exploit some component-specific characteristics. While these do not gain the full benefits, they are simple to exploit and represent early steps toward component-specific mapping.

12.4.1 Component Matching

In its EasyPath program, Xilinx tolerates defects in fabricated FPGAs by matching the needs of a particular design to particular fabricated components [49, 58]. Since no design uses all of the features and resources in an FPGA, defects in the unused resources are tolerable. Given a customer design, Xilinx simply needs to make sure that they identify particular FPGAs where the defects do not interfere with the user's logic. This avoids additional work to map for a specific component. Using design-specific testing [89], this approach also avoids the need to create a map of the defects on the FPGA. However, the components assigned to a design may not be capable of supporting any changes that may be needed to the design.

12.4.2 Multiple Bitstreams

Since there is never a single way to map a design to an FPGA, we can exploit this freedom to avoid defective or undesirable devices. At a coarse granularity, one could place and route a design several times to produce multiple bitstreams and then test the bitstreams on the component. If any of the bitstreams avoids all the defective devices, we have a successful mapping [60, 77, 87]. With care, one can generate a set of complementary mappings that deliberately use distinct resources. For example, if no single resource is used in every bitstream in a set, we are guaranteed to be able to avoid any single defective resource. This technique also avoids the need for per-component mapping and the need to generate a defect map at the cost of performing design-specific testing. This monolithic scheme is viable when there are just a few defective resources in an FPGA but does not scale well to high defect rates and high variation.

12.4.3 Body Biasing and Dual V_{dd}

In (Chapter 10, Section 10.4.5 and 10.4.6) we saw that dual V_{dd} and body biasing can be used to reduce power dissipation. Both techniques operate on similar principles: a non-critical block of logic can be assigned a lower V_{dd} to reduce dynamic power, or it can be assigned a higher V_{th} through body biasing to reduce static power.

Both techniques will reduce the speed of the block but save power. Blocks that contain critical paths must be assigned either a high V_{dd} or low V_{th} to maintain performance. Because the primary impact of variation is that some devices become fast and leaky while others become slow and less leaky, these same techniques can be used to improved both timing yield and leakage yield.

Nabaa et al. examined body biasing for an FPGA to improve timing yield [62]. In their scheme each CLB and switchbox is assigned a body bias circuit. A single on-chip characterization circuit measures the delay of each block one at a time by comparing the expected block delay to its actual delay using a phase detector. The circuit then sets a three bit register local to the block that determines the applied bias voltage. They find that for a 130 nm technology (with unspecified variation) body biasing can reduce timing variation by 3.3× and leakage variation by 18×.

Bijansky et al. studied the impact of tuning dual V_{dd} FPGA architectures to compensate for variation [11]. In their architecture each CLB can choose from two voltages, with voltage assignment taking place post-fabrication on a component-specific basis. They assume some form of delay measurement to provide the information necessary for their voltage assignment algorithm. For a 65 nm predictive technology with $3\sigma_{V_{\mathrm{th}}}/\mu_{V_{\mathrm{th}}} = 20\%$ random variation, they report an average timing yield improvements of 15%.

The primary drawback of these techniques is that they have limited granularity–they can only impact the V_{dd} or V_{th} of block, not of individual transistors. While they are ideal for region-based variation, these techniques do not scale well with high random variation; supporting individual V_{dd} connections or bias circuitry on a per transistor basis would add excessive area overhead, defeating the benefits offered by this tuning mechanism.

12.4.4 Post-Silicon Clock Tuning

Component-specific mapping concepts have also been incorporated into ASIC designs, such as the Itanium 2 processor [86]. Process variation for high speed ASIC designs have a significant impact on critical path delays and clock skews that can lead to timing yield loss. The Itanium 2 includes programmable delay buffers in the clock distribution network that can be configured after manufacturing to compensate for these delay variations. Clock buffers at the second lowest level of the clock tree contain 32 different delays that can be selected by programming a 5-bit register. Clock skew can be compensated by selecting the appropriate delays in the skewed portion of the tree. Critical path delay can also be improved – by slowing down/speeding up clock signals to the input/output registers of a critical path, time can be borrowed from adjacent paths (if those paths are not also critical). Thus, the clock period of the entire chip can be shorter than the worst-case path. Currently, Intel programs these buffers on a per production stepping basis rather than on a component-specific level to reduce production test time, but in principle this technique could be applied per-device. Algorithms for automatically tuning delay buffers such as these on a per-chip basis is a subject of active research (e.g., [64]).

Post silicon tuning has also been demonstrated for FPGA clock networks [79] but at a fully component-specific level. Programmable delay buffers are inserted at the leaf of the clock tree at every flip-flop clock input. Delays are assigned by measuring certain designated paths using techniques from [76]; these measurements are then used to predict actual critical path delays through statistical analysis. This enables approximate clock skew assignment without having to obtain full delay knowledge.

12.5 NanoPLA

Bottom-up assembly of nanoscale building blocks is a promising option for continued scaling, avoiding the need for lithographic patterning at the single-digit nanometer scale. However, bottom-up assembly techniques are limited to extremely regular structures and are prone to high defect rates. Consequently, FPGA-like post-fabrication configuration will be essential for these components.

The NanoPLA [26] is a nanoscale reconfigurable system built out of silicon nanowires that may have a mean transistor channel length of 5 nm or smaller. This section explains how the NanoPLA is assembled, how it is used for computation and where it will exhibit defects and variation. Section 12.6.3 summarizes the technique, presented in [36], that successfully manages the problem of high variation in the NanoPLA transistors. Later, in Section 12.7.2, we highlight a component-specific approach, presented in [30], to crosspoint defect tolerance for this architecture.

12.5.1 Technology: Nanowires

Nanowires are the main building block of the NanoPLA. These can be grown out of many different materials including doped Si [23], GaAS, GaN [70], and Au [88]. These wires can be microns long [42] and their diameters can be precisely controlled using seed catalysts [23]. Moreover, during the growth process, the doping of the nanowire can be varied along its length [41, 98], allowing components such as field-effect transistors to be embedded in the wire. Finally, insulating core shells can be radially grown over the entire length of the wire, creating a separation between conducting wires as well as between gate and control wires in a FET [52, 53].

Due to their small features and limited assembly techniques, regular structures are easier to build out of these components than arbitrary topologies. Langmuir-Blodgett (LB) flow techniques are used to align nanowires into large-scale parallel arrays [47, 90]. By using nanowires with insulating shells, the LB technique can tightly pack nanowires without shorting them. These shells can later be selectively etched away [90]. When repeated, this process allows for two orthogonal layers to form a densely packed nanowire crossbar [47, 90].

Furthermore, chemist have demonstrated a number of techniques for placing hysteretic switches into the crosspoints between orthogonal nanowire layers. These include layers of bi-stable molecules [18, 16], amorphous silicon nanowire coatings [32], nanowires made of switchable species [34], and memristive technologies [13]. Some of these programmable switches have diode-like rectification, an essential property for correct operation.

12.5.2 Architecture: NanoPLA

The NanoPLA is organized as shown in Fig. 12.11b. It consists of tiled logic blocks with overlapping nanowires that enable Manhattan routing while maintaining direct nanoscale-density interconnect among blocks. It is introduced in [37] as a modification on DeHon's [26] NanoPLA blocks and uses amorphous Si switches [32] to improve performance and energy.

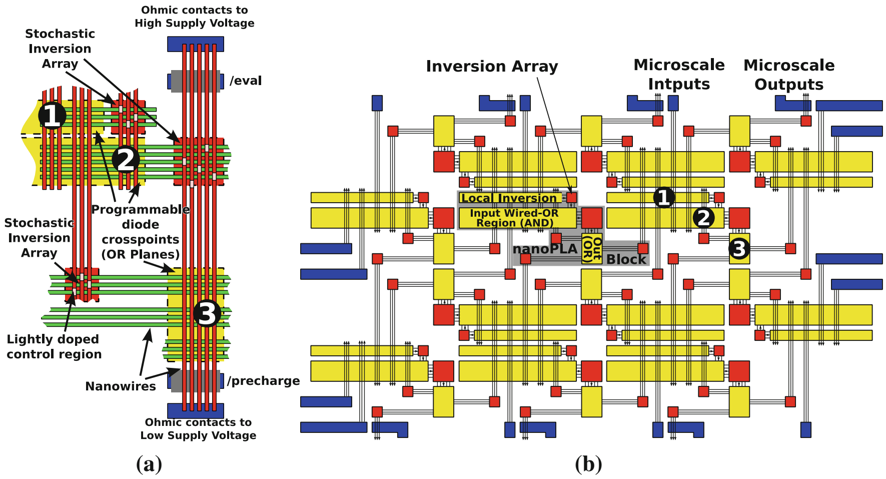

Ohmic contacts to High Supply Voltage
Stochastic Inversion Array
/eval
1
2
Programmable diode crosspoints (OR Planes)
Stochastic Inversion Array
3
Lightly doped control region
Nanowires
/precharge
Ohmic contacts to Low Supply Voltage
(a)
Inversion Array
Microscale Intputs
Microscale Outputs
Local Inversion
Input Wired-OR Region (AND)
nanoPLA
Out (OR)
Block
1
2
3
(b)

Fig. 12.11 NanoPLA organization. (**a**) NanoPLA block; (**b**) NanoPLA block tiling

The NanoPLA block is composed of three logic stages. As in a conventional PLA, the first stage or input stage is used to selectively invert the inputs ❶. Stage two and three behave like the AND ❷ and OR ❸ planes, respectively. Adding the initial inverting phase allows us to avoid the need for non-inverting restoration as used in [26] and thus improve the overall performance and reduce the total energy used.

Figure 12.11a shows a detailed view of a NanoPLA block. Using the bottom-up assembly discussed above, small diameter nanowires are arranged into tight-pitch parallel arrays. Though logically each plane performs a different function (Invert, AND, and OR), physically all three planes are identical and are made up of a diode-programmable, wired-OR stage built using the switches previously described, followed by an inversion stage where lightly doped regions of the nanowire behave like field-effect gates and provide restoration. During assembly, etching is used to differentiate the three stages. Not shown in Fig. 12.11a are the nanowire decoders used to program the diode-like switches. They are built as described in [26, 29] and demonstrated in [98].

The NanoPLA is similar to conventional FPGAs. Both use Manhattan routing to connect discrete clusters of logic. However, routing in the NanoPLA is done through the blocks rather than using an independent switching network. In order to allow signal routing, the output of the OR-plane of every block connects to itself and four neighboring blocks as shown in Fig. 12.11b.

12.5.3 Defects and Variation

Initially, due to its manufacturing, and later, during operation as it ages, the NanoPLA will experience many defects as well as extreme variation. However, due to the reconfigurable nature of the NanoPLA, a design mapped to it can avoid the defects (Section 12.7.2) and successfully use the variation in the system (Section 12.6.3, [36]).

Since the nanowires and programmable nanowire–nanowire junctions are the main nanoscale components of the NanoPLA, we expect two major nanoscale defect types: broken wires and defective junctions. Within the latter, the technology tends toward non-programmable junctions rather than stuck-on junctions.

Unlike today's technology where region-based and systematic variation dominate, in the NanoPLA random variation dominates due to the bottom-up manufacturing process. Along with the variation that affects even today's technology (*e.g.* local oxide thickness variation [6], statistical dopant variation [5] and dopant placement, line-edge roughness [7], and channel length variation [85]) the NanoPLA faces additional sources of *random* variation, including the following:

- Nanowire geometries and features (e.g., length of doped regions, core shell thickness) will vary independently since each nanowire will be individually fabricated.
- Statistical alignment techniques [99] during assembly cause the geometry of the field-effect regions to vary from device to device [29].
- Each programmable diode region will be composed of a small number of elements or bonds, giving them large, random variation from crosspoint to crosspoint.

These sources of variation manifest as differences in the nanowire resistances and capacitances, the diode resistances, and the threshold voltages (V_{th}) for the field-effect restore nanowires. All of these variation effects can be modeled by independent Gaussian distributions with mean μ and standard deviation σ (e.g., [5, 6, 7, 85]).

$$P(x) = \left(\frac{1}{\sigma\sqrt{2\pi}}\right) \mathrm{e}^{\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)} \tag{12.4}$$

Throughout this chapter the amount of variation is expressed as a percentage equal to σ/μ; from here on referred to simply as σ. Though other works also report variation as a percent, it is worth noting that many, including the ITRS [1], tend to report 3σ variation.

12.5.4 Nanoscale Memory and Crossbar Arrays

Several groups have proposed nanoscale PLA, interconnect, and memory architectures built around emerging nanoscale technologies that share many properties with the NanoPLA architecture (see Table 12.2) [28, 44 Chapter 38]. All are based on tight-pitched, crossed arrays of nanoscale wires. They almost all use some form of diode-programmable crosspoint and perform wired- OR logic on the nanoscale wires. The specific proposals employ a variety of fabrication techniques, programmable crosspoint technologies, and nanoscale addressing techniques. Nonetheless, due to their nanoscale features, they all have challenges with high variation and defect rates similar to those encounted by the NanoPLA. While the solutions we elaborate here are evaluated concretely for the NanoPLA, they will have similar benefits for this broad class of crossbar-based architectures.

12.6 Mapping with Full Knowledge of Component Characteristics

We now look at several cases of component-specific mapping. First, we review work on component-specific placement for region-based variation in FPGAs (Section 12.6.1). Then we examine the potential for energy reduction in FPGAs with component-specific routing for random variation (Section 12.6.2). In Section 12.6.3, we explore mapping for the NanoPLA in the presence of extreme transistor variation. Finally, Section 12.6.4 examines the measurement precision to support the component-specific mapping for the NanoPLA.

12.6.1 Placement

Some existing works of Cheng et al. [19] and Katsuki et al. [48] have explored modifying the FPGA CAD flow to incorporate component-specific mapping using full delay knowledge. These schemes use delay knowledge during placement to place critical paths in fast chip regions and non-critical paths in slow regions (Fig. 12.12). Every chip receives a unique placement and is then routed without delay knowledge.

Table 12.2 Comparison of nanoelectronic programmable logic designs

Component element	HP/UCLA Xbar arch.	CMU nanofabric	NanoPLA	Stony brook CMOL	HP FPNI	CASE MBC
Crosspoint technology	program diode	program diode	program diode	program diode	program diode	program diode or STTRAM
Nanowire technology	nanoimprint lithography	nanoPore templates	catalyst nanowires	nanoimprint lithography	nanoimprint lithography	nanoimprint lithography
Logic implementation	nanoscale wired-OR	nanoscale wired-OR	nanoscale wired-OR	nanoscale wired-OR	lithoscale (N)AND2	memory
CMOS↔Nano interface	random particles	–	coded nanowires	crossbar tilt	crossbar tilt	random particles
Restoration	CMOS	RTD latch	nanowire FET	CMOS	CMOS	CMOS
References	[45, 59, 92]	[38, 39]	[25, 31]	[84]	[81]	[66, 67]

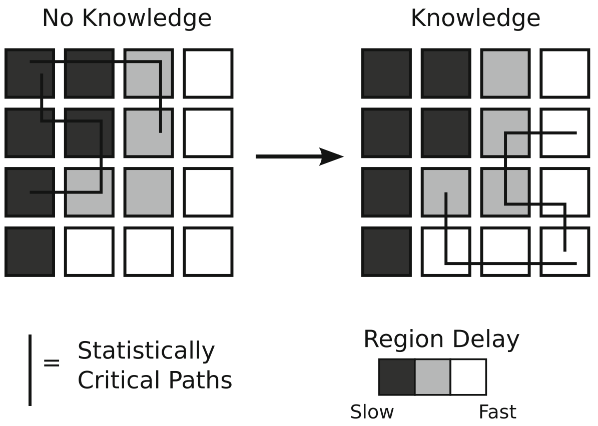


Fig. 12.12 Full knowledge placement for region-based variation

To perform chipwise placement, a variation map must be generated on a per chip basis. A critical assumption of this technique is that an FPGA can be divided into regions (typically sized as a CLB plus surrounding routing resources) where each region has similar performance characteristics. Regional delay characteristics can be obtained by techniques covered in Section 12.3, where test circuits are mapped to each region and measured through at-speed path delay tests. Once the variation map is obtained, placement proceeds precisely as in the knowledge-free case, except instead of assigning all resources the same delay in the placement algorithm, resources within a region are assigned a delay from the variation map.

Katsuki et al. custom fabricated 31 90 nm FPGA test chips and constructed a delay map of each chip [48]. Under an unspecified amount of variation, they demonstrated a delay improvement of 4.1%. Cheng et al. performed a similar knowledge placement experiment under simulation [19]. For $3\sigma/\mu = 10\%$ variation in both L_{eff} and V_{th}, they report on average between 6.91 and 12.10% performance improvement.

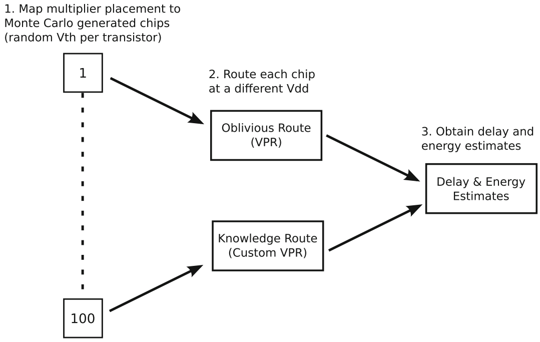
An important limitation of component-specific placement is the assumption that variation is primarily due to the spatially correlated component (regional) rather than the uncorrelated, random component. Variation in future technologies is expected to be dominated by random effects; hence, there will a stronger effect from fast or slow *resources* rather than fast or slow regions. Hence, it is also necessary to address variation at a lower level than regions.

12.6.2 Routing

While placement with full delay knowledge can compensate for spatially correlated variation, routing can more effectively deal with random variation. Routing in reconfigurable devices enables fine-grained resource selection and avoidance, which is critical in future technologies where neighboring devices can have drastically different characteristics (Chapter 1, Section 1.2.3). Compared to a traditional, one-mapping-fits-all routing scheme, routing with full delay knowledge can improve delay and reduce energy margins. This section introduces preliminary results for delay and energy savings achievable from component-specific routing on a conventional FPGA architecture.

To perform component-specific routing on an FPGA, it is necessary to first measure and store the delay of every switch in a device (Section 12.3); this delay map can then be used by the router. We modified the router in VPR 5.0 [2], a standard academic FPGA CAD tool, to use resource delays from a delay map rather than fixed values. The routing algorithm in VPR is based off of PathFinder [61]; we make additional optimizations to increase router stability and reduce experimental noise. The PathFinder algorithm iteratively routes paths via the A* least-cost path algorithm until a solution is found. For timing critical paths the cost of a resource is determined by its delay. By modifying the router to use actual, measured delays instead of static values, critical paths will be mapped to known low cost, fast resources, improving overall delay.

Fig. 12.13 Experimental flow


1. Map multiplier placement to Monte Carlo generated chips (random Vth per transistor)
1
100
2. Route each chip at a different Vdd
Oblivious Route (VPR)
Knowledge Route (Custom VPR)
3. Obtain delay and energy estimates
Delay & Energy Estimates

The full experimental flow for our delay-aware routing experiments is shown in Fig. 12.13. As a basis for demonstration we used an optimized, fully pipelined 16-bit multiplier as our benchmark circuit (Fig. 12.14). A single, hand-generated placement is mapped to 100 chip instances, each with a unique V_{th} variation map. The delays in the variation map are generated using delay models from [16] and correlated to HSPICE delays generated using the 22-nm low power Predictive Technology Models (PTM) [100] with $V_{\text{th}} \approx 550$ mV. All 100 chips are routed both with delay-oblivious and delay-aware routers at several values of V_{dd}. We route on an FPGA architecture with clusters made of four 4-LUTs, length 4 segments, bidirectional subset switches, and with twice the minimum channel width to provide sufficient extra segments for resource avoidance.

Figure 12.15 plots delay for 90% parametric yield over 100 chips as a function of V_{dd} for the delay-oblivious and delay-aware routers at $\sigma_{V_{\text{th}}}/\mu_{V_{\text{th}}} = 0\%$, 15%, and 30% (ITRS predicted at 22 nm). 90% parametric delay is calculated as the 90th/100

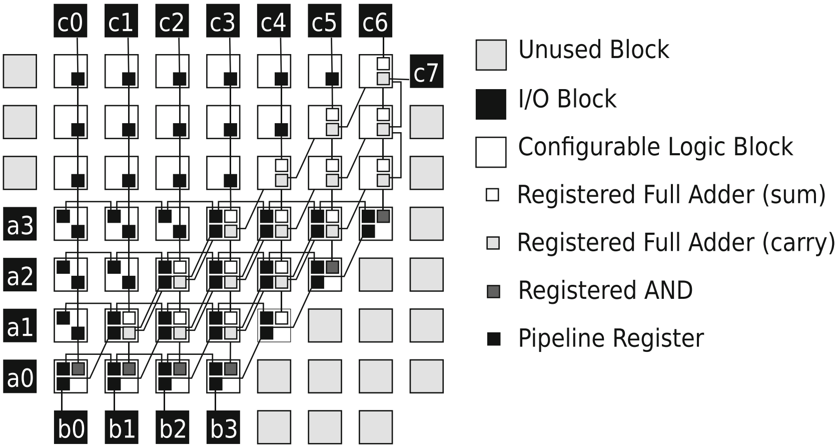


Fig. 12.14 Pipelined FPGA multiplier design (4-bit example)

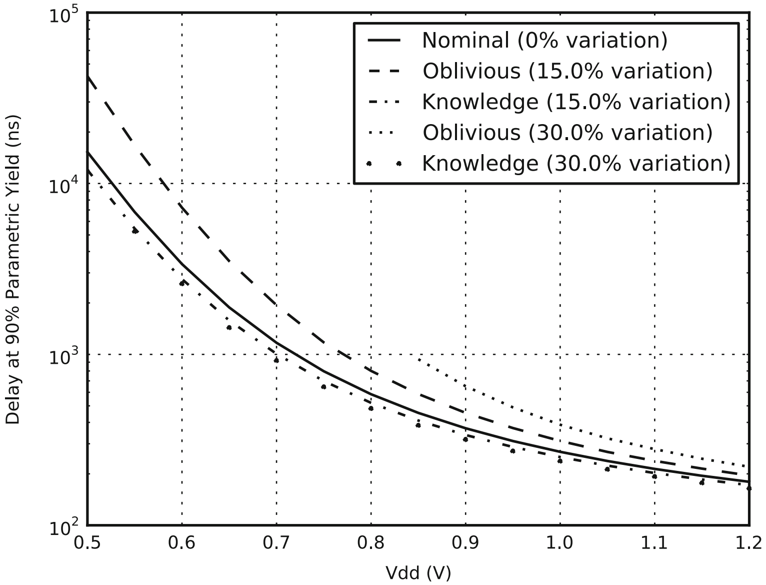


Fig. 12.15 Delay vs V_{dd} for full delay knowledge and delay-oblivious routers under variation

fastest chip. As variation increases, we see that the delay of the delay-oblivious router increases substantially over the 0% variation case. Additionally, at 30% variation the delay-oblivious design cannot function under 850 mV without encountering switching defects. The full delay knowledge router, however, can function across all voltages and produces significantly faster designs. In fact, the delay-aware router produces designs that are *faster* than those with no variation. When compared to the delay-oblivious router, the full delay knowledge improves performance by at most

3× at 15% variation (500 mV) and 30% variation (850 mV). We also see that if a fixed performance is needed, the delay-aware router can achieve the same performance at a lower V_{dd} when compared to the delay-oblivious router. For example, at 30% variation and a delay target of 1000 ns, the delay-oblivious router requires $V_{\text{dd}} = 850$ mV while the full delay knowledge router can achieve the same performance at $V_{\text{dd}} = 700$ mV, a savings of 150 mV.

To quantify this improvement in energy at a fixed performance, Fig. 12.16 plots energy/operation as a function of 90% parametric delay. Energy/operation is obtained from the 90th/100 fastest chip. We see that for 15% variation delay-aware routing continually reduces energy/operation as we increase the target delay. For 30% variation, delay knowledge enables significant energy savings for high performance requirements but these savings diminish with slower targets. Figure 12.17 plots the energy/operation ratio of delay-oblivious to delay-aware routing. Here we see that full delay knowledge routing can achieve a maximum energy reduction of 2.2× for 15% variation and 3.1% variation at their ideal delay targets.

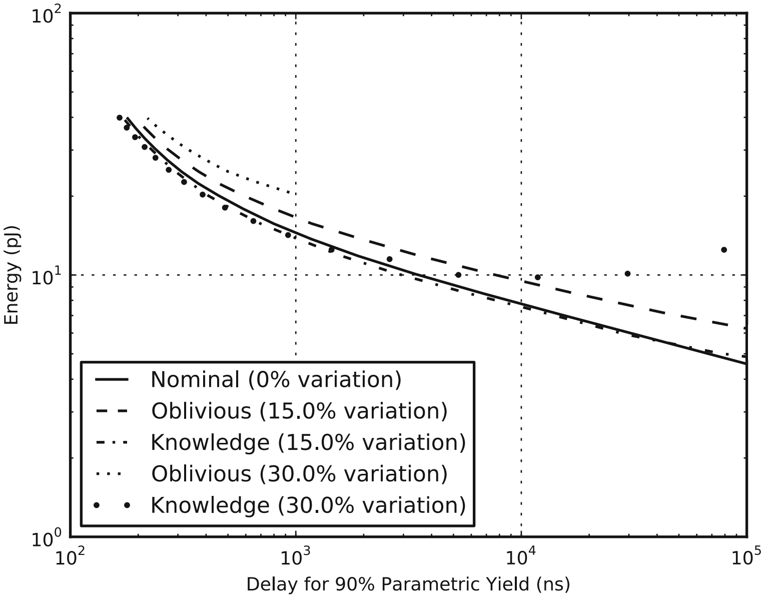


Fig. 12.16 Energy vs delay target for full delay knowledge and delay-oblivious routers

These initial results show the promise of reducing energy operation by using component-specific mapping for conventional FPGAs. However, these results are still preliminary and have only explored a single benchmark circuit and a single FPGA architecture. In future work we will expand our experiments to include a wide set of benchmark circuits, an array of architecture parameters (e.g., switchbox type, segment length, switch directionality), different values of extra channels, and further algorithmic optimizations (e.g., optimizing VPR to target energy reduction directly).

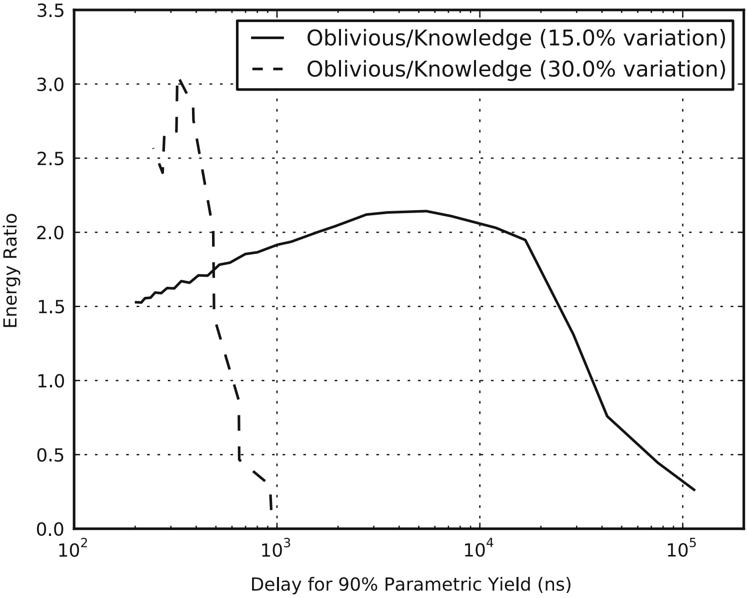


Fig. 12.17 Energy ratio of full delay knowledge to delay-oblivious routing vs delay target

12.6.3 NanoPLA Variation Matching

Previous sections highlight how component-specific mapping can be used on conventional FPGAs to reduce power consumption or increase performance. For modest variation, if component-specific mapping is not used, it will cost energy or delay but the circuit will still work. This is not the case for the NanoPLA. Between its expected high variation and the use of precharge logic, designs mapped oblivious to variation are highly likely to fail completely.

Due to the bottom-up fabrication of the NanoPLA, random variation is the predominant form of variation. Although the wires and programmable junctions will exhibit variation, the exponential dependence of leakage current on threshold voltage means the dominant variation is in the restoring transistors. The apparent off resistance of the transistors in the NanoPLA will vary by approximately ten orders of magnitude. A design mapped oblivious to variation will be highly likely to use some resources far out in the tails of this very wide distribution. When this happens, the mapped circuit will be unable to meet timing (regardless of how slow or fast the timing is) and will fail. The problem is aggravated by the logical variation in the form of varying fanout of the nets being mapped. Functions mapped to the NanoPLA will have varying amount of downstream capacitance proportional to their fanout. This is due to the fact that the NanoPLA uses programmable non-restoring, diode-like connections to implement these functions. On average, fanout varies in the range of two orders of magnitude; combining with the ten orders of magnitude from the transistor, this leads to a delay varying over 12 orders of magnitude. Full knowledge, component-specific mapping on the NanoPLA, however, can carefully select resources from the distribution of the apparent off resistance of the transistor and match them with functions so that the fanout of the function combines with the transistor's variation to *reduce* the overall variation. This is necessary for the NanoPLA to function.

In this section, we will see why a variation oblivious mapping fails and understand what causes this failure. Once we understand why the failure happens, we will review VMATCH [35, 36], a mapping algorithm that is adapted to prevent this failure by using the logical variation to counteract the physical variation. Finally, we show sample VMATCH results to illustrate its behavior in practice.

12.6.3.1 Variation Oblivious Mapping

The NanoPLA is clocked at a very fine-grained level. Each resource between clock boundaries is called a NAND-*term* because it computes the equivalent of a multi-input NAND gate. Figure 12.18 shows a nanowire diagram and circuit representation of a NAND-term. It is composed of a transistor operating as a precharge inverter followed by a section of programmable diodes computing a wired-OR.

When the input is low, the input transistor allows charge to flow through to the programmable diodes and to the output terminal. The time it takes to charge the output when the input transistor is on, is called τ_{switch}. Similarly, when the input is high, the transistor will be in its off state. The time it takes the output terminal to charge due to the transistor leaking in the off state is the τ_{leak} of the NAND-term.

A design mapped to a NanoPLA will fail if two resources, among the resources used, are such that the τ_{leak} of one is *less than* the τ_{switch} of the other. In other

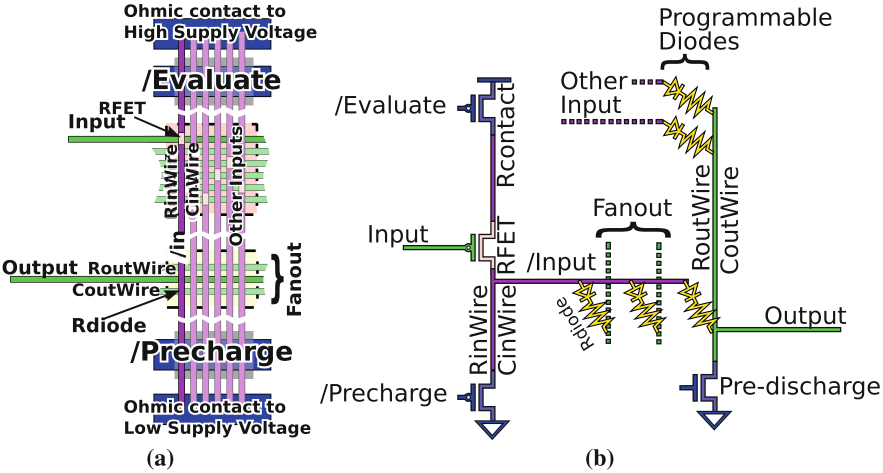


Fig. 12.18 NanoPLA NAND-term. (**a**) Physical implementations (**b**) Circuit diagram

words, there is a resource that leaks faster than another resource can switch. A design mapped oblivious to variation is highly likely to choose two such resources.

From the failure condition above it follows that in order for a design to work successfully on the NanoPLA, the τ_{leak} of *all* resource must be greater than the τ_{switch} of all resources. In practice, to keep leakage to within 1% of the total switching energy, we require that there is at least two orders of magnitude separation between the NAND-term with the slowest τ_{switch} and the NAND-term with the fastest leakage time. Equation (12.5) captures this requirement.

$$100 \cdot \max(\tau_{\text{switch}}) \leq \min(\tau_{\text{leak}}) \tag{12.5}$$

To see if a mapping was successful, we can plot, on a log scale, the distribution of the leakage and 100 times the switching time (100 to account for the two orders of magnitude separation required) for all used NAND-terms. If there is a separation between the two distributions, then Equation 12.5 will hold and the mapping will be successful. Figure 12.19 plots these distributions for the variation oblivious mapper for a particular circuit at $\sigma = 38\%$ variation. It is readily apparent that this mapping failed since there is no separation between the two distribution. Moreover, the distribution of τ_{leak} is about 12 orders of magnitude wide.

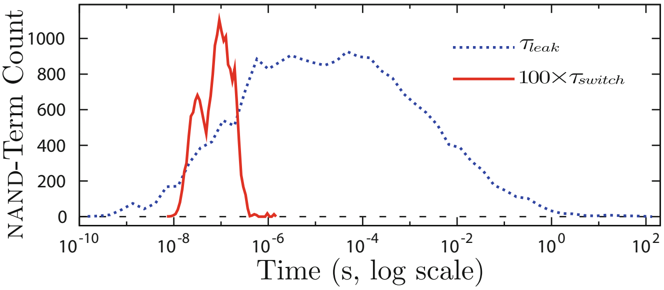


Fig. 12.19 Distribution of τ_{leak} and $100 \times \tau_{\text{switch}}$ of a delay oblivious-mapping at $\sigma = 38\%$

There are two effects contributing to the width of this distribution. First and foremost is the physical variation in the devices of the NanoPLA. The second cause is logical variation in the circuit being mapped. The physical variation in the NanoPLA has a direct effect on the electrical properties. Specifically, it affects the diode resistance, wire resistance and capacitance, and transistor threshold voltage. Equations (12.6 and 12.7) show how τ_{switch} and τ_{leak} depend on device characteristics.

$$\begin{aligned}\tau_{\text{switch}} = &\left(R_{\text{contact}} + R_{\text{onFET}} + \frac{1}{2}R_{\text{inWire}}\right) \times \left(C_{\text{inWire}} + \sum_{\text{fanout}} C_{\text{outWire}}\right) \\ &+ \left(R_{\text{diode}} + \frac{1}{2}R_{\text{outWire}}\right) \cdot C_{\text{outWire}}\end{aligned} \tag{12.6}$$

$$\tau_{\text{leak}} = \left(R_{\text{contact}} + R_{\text{offFET}} + \frac{1}{2}R_{\text{inWire}}\right) \times \left(C_{\text{inWire}} + \sum_{\text{fanout}} C_{\text{outWire}}\right) + \left(R_{\text{diode}} + \frac{1}{2}R_{\text{outWire}}\right) \cdot C_{\text{outWire}} \tag{12.7}$$

The only difference between τ_{switch} and τ_{leak} is the state of the transistor being on (R_{onFET}) and off (R_{offFET}), respectively. R_{offFET} is the apparent resistance of the transistor in the sub-threshold region or $R_{\text{offFET}} = V_{\text{dd}}/I_{\text{sub}}$ (Equation (12.9)). In the on state, the transistor operates in saturation, and we define the value of R_{onFET} as $V_{\text{dd}}/I_{\text{sat}}$ (Equation (12.8)). Since the nanowires are still silicon, we use short-channel P-Type MOSFET current equations [43, 69]:

$$I_{\text{sat}} = W v_{\text{sat}} C_{\text{ox}} \left(V_{\text{th}} - V_{\text{gs}} - 0.5 \cdot V_{\text{d,sat}}\right) \tag{12.8}$$

$$I_{\text{sub}} = \frac{W}{L}\mu C_{\text{ox}}(n-1) \cdot {v_T}^2 \mathrm{e}^{\frac{V_{\text{th}}-V_{\text{gs}}}{n v_T}} \left(1 - \mathrm{e}^{-V_{\text{ds}} \cdot {v_T}^{-1}}\right) \tag{12.9}$$

The width of the τ_{switch} and τ_{leak} distributions can now partly be explained by the fact that saturation current is *linear* in V_{th} and V_{dd} and that sub-threshold current is *exponential* in V_{th} and V_{dd}. Thus a small change due to the variation in V_{th} will cause a *linear change* in the value of R_{onFET} and an *exponential change* in the value of R_{offFET}. At $\sigma = 38\%$, the 3σ V_{th} variation point gives a range for R_{offFET} from $1.8 \times 10^7 \Omega$ to $7.0 \times 10^{16} \Omega$. Nonetheless, this physical variation only accounts for 10 out of the 12 orders of magnitude spread of the τ_{leak} distribution; the remaining two orders come from the logical variation in the design being mapped.

Along with the physical parameters of Equations (12.6) and (12.7), there is a fanout parameter (the summation over *fanout*) whose value comes directly from the logical netlist and varies over a significant range. Fanout in the NanoPLA comes from the fact that a NAND-term has non-restoring, diode-like connections (Fig. 12.18). If a signal on an input wire is needed by multiple output wires, the input wire must have the associated diodes programmed to connect to the required output wires, and it must charge up all connected wires. Consider an example: when mapping the logical function $A\overline{B} + AC\overline{D} + B\overline{E} + AF$ to a block in the NanoPLA, three terms in the AND-plane will use input signal $A(A\overline{B}$, $AC\overline{D}$, and $AF)$, while signal F is only used once by AF. Even without physical variation, this means that signal A's NAND-term will see three times the capacitive load that F's will. Figure 12.20 shows a typical fanout distribution for a NanoPLA with 64 NAND-terms and length-2 segments. Figure 12.20 shows a maximum fanout of 34. While there are a few high fanout nets, note that most of the nets have fanout one. Mapped obliviously, this adds another two orders of magnitude to the range of the τ_{leak} distribution; this makes fanout the second-most significant source of variation in the NanoPLA.

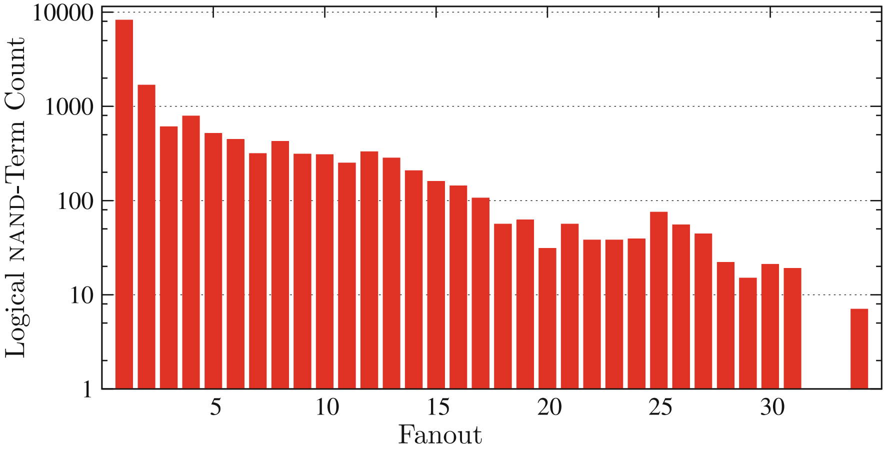


Fig. 12.20 Fanout distribution

12.6.3.2 VMATCH: NanoPLA Variation Aware Mapping

VMATCH is a mapping algorithm that reduces the overall variation by using the logical variation from the fanout to counteract the physical variation, mainly of the transistor. It does this by carefully matching function with high fanout to transistors with low apparent resistance and vice versa. This matching allows VMATCH to both reduce the spread of the τ_{leak} distribution and to use resources that would otherwise have been considered to be too leaky.

To perform this post-fabrication, variation-aware mapping, it is necessary to measure the nanowire transistor threshold voltage. We take advantage of the NanoPLA architecture to configure voltage dividers between each input wire and a reference resistance to estimate the input wire's resistance and in turn the transistor's off resistance. Section 12.6.4 briefly explains how this measurement is done. A full analysis of the measurement technique is presented in [35].

With this information, VMATCH can correctly match functions to resources. The algorithm is a two-step process that first performs an analysis to determine if a mapping is possible given the placement and global-route computed. This analysis determines the worst-case delay, $\tau_{\text{switchFeasible}}$, achievable for the given design. If a mapping is possible, VMATCH individually maps each NanoPLA plane by matching function to resources so that the mapping meets or exceeds the $\tau_{\text{switchFeasible}}$ target calculated in the first step's analysis. By first finding $\tau_{\text{switchFeasible}}$ and then mapping each plane, the algorithm makes sure that not only will each plane meet the two orders of magnitude separation required, but over all planes, this separation is guaranteed.

To compute the worst-case delay, $\tau_{\text{switchFeasible}}$, the algorithm finds the slowest possible mapping for each plane. $\tau_{\text{switchFeasible}}$ is then the maximum τ_{switch}, over all planes, resulting from this slow mapping. For each plane, the slow mapping is computed by matching the slowest resource to the function with the lowest fanout, the second slowest resources to the function with the second lowest fanout and so on until all functions have been mapped. The exponential nature of τ_{leak} as V_{th} varies, compared to the linear behavior of τ_{switch}, means that this slow mapping is also the mapping with the largest separation between τ_{switch} and τ_{leak}. A large separation increases the probability that a separation of two orders of magnitude will be possible over all planes. If a slow mapping in one plane does not achieve two orders of magnitude, then the mapping will fail for the NanoPLA since a faster mapping for that plane will only decrease the separation. If a failure does occur, we can resolve it by either adding more resources or by redoing placement and/or global route with the knowledge of how this plane failed.

Once $\tau_{\mathrm{switchFeasible}}$ is calculated, if the mapping is possible, VMATCH maps each plane individual. To map a plane, we could compute the bipartite graph with an edge between a resource and a function if and only if the mapping of that function to that resources produces a $\tau_{\mathrm{switch}} \leq \tau_{\mathrm{switchFeasible}}$ and a $\tau_{\mathrm{leak}} \geq 100 \times \tau_{\mathrm{switchFeasible}}$. We could give a weight to each edge equal to the negative τ_{switch} resulting from the mapping represented by that edge and then we could solve the maximum correspondence maximum value bipartite matching to get the fastest solution. However, by sorting functions from high fanout to low fanout, and resources from fast to slow, we can greatly simplify this process and compute a greedy matching that achieves the same result. After sorting, starting with the highest fanout function, we match it to the fastest resource that meets the requirement $\tau_{\mathrm{switch}} \leq \tau_{\mathrm{switchFeasible}}$ and $\tau_{\mathrm{leak}} \geq 100 \times \tau_{\mathrm{switchFeasible}}$. If a resource is rejected, it is never considered again since a function with lower fanout would only find that resource even faster than this function did. When a matching is found, we continue with the next highest fanout function, searching over the remaining resources. At the very least, this process is guaranteed to find the slowest solution from part one of the algorithm. Yet, by starting with the fastest resources, we can often find significantly faster solutions. Figure 12.21 shows the result of running VMATCH on the same circuit as the one used for Fig. 12.19.

To see the effect that VMATCH has on resource selection, compare the distribution of the apparent off resistance of the transistor for the oblivious mapping and the mapping generated by VMATCH (Fig. 12.22). The oblivious algorithm chooses resources across the whole distribution, while VMATCH mostly uses fast resources but never uses resources that are too leaky. Thus, by using full knowledge, component-specific matching, VMATCH, not only guarantees that the mapping will not be defective by maintaining a separation between the leakiest and slowest resource, but also exploits the fast resources, thus improving performance.

12.6.3.3 VMATCH Quality and Yield

VMATCH is necessary if the NanoPLA is to function correctly even at lower variations than the ITRS [1] predicted 38% variation for 5-nm transistor channel lengths. Figure 12.23 compares the yield of VMATCH with that of the variation oblivious

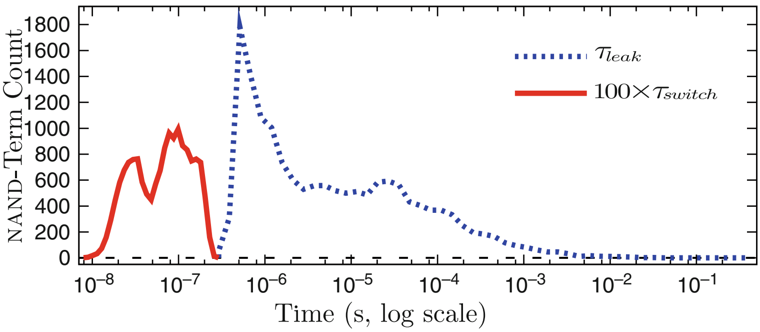


Fig. 12.21 Distribution of τ_{leak} and 100 × τ_{switch} of VMATCH mapping

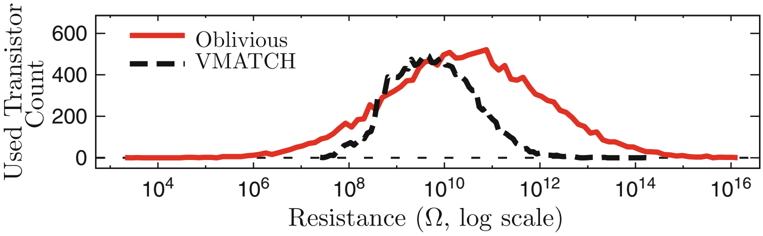


Fig. 12.22 Used R_{offFET} distribution for oblivious and VMATCH algorithms

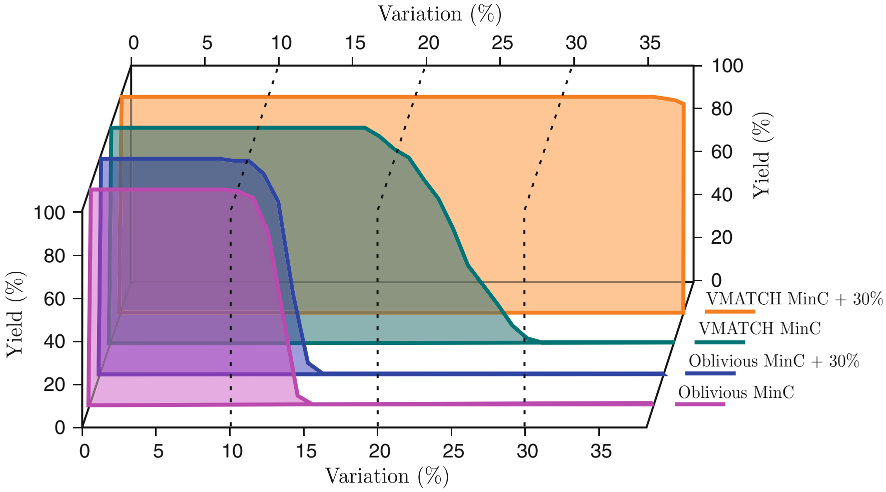


Fig. 12.23 Yield curves for oblivious and VMATCH mapping at minimal channel width and with 30% extra channels

mapper as variation increases from variation free to 38%. Two architecture configurations are examined, routing on the minimum channel width, minC, (i.e., no spare NAND terms for the arrays with the most logical NAND terms) and routing with 30%

extra resources above this minimum. For each variation point, all four configurations were tested on 100 Monte Carlo generated chips; this means that with 90% confidence, the results estimated as 100% yield at least exceed 97.5% yield.

Up to, approximately 9% variation, 100% yield is achieved by all mapping-configuration combinations, but as variation increases, the oblivious mapper quickly falls and at 15% variation it no longer is able to yield. VMATCH, in contrast, maintains 100% yield long beyond this point. At minimum channel width, VMATCH starts to gradually loose yield. Since VMATCH carefully chooses which resources to use, it can effectively use the 30% extra resources to maintain full yield at high variation.

Table 12.3 Percent of extra resources beyond minimum channel width required to achieve 100% yield on 100 chips at 38% variation

	% Extra channels			% Extra channels	
Net	Mean	St.dev.	Net	Mean	St.dev.
alu4	16	7	ex5p	15	7
apex2	21	11	frisc	40	18
apex4	14	8	misex3	19	9
bigkey	14	11	pdc	21	7
clma	32	17	s298	19	9
des	21	10	s38417	26	12
diffeq	26	15	s38584.1	31	13
dsip	27	12	seq	16	6
elliptic	20	12	spla	16	6
ex1010	32	14	tseng	25	12

Table 12.3 shows the percent of extra resources needed to successfully map the Toronto 20 benchmark set [10] using VMATCH. On average, 22% extra resources above the minimum channel width are required. This increase in resources will reduce the overall performance and increase energy and area as compared to the variation-free case. Nevertheless, VMATCH designs are not much larger, slower, or energy hungry than the nominal case.

Along with VMATCH, [36] presents a sparing-based mapping algorithm that thresholds highly leaky resources, marks them as defective, and routes using the remaining resources. Compared to the variation-free case, this algorithm uses 2.6 times more energy, making this algorithm too expensive. In contrast, VMATCH, on average only uses 20% more energy. Table 12.4 compares VMATCH's energy use to that of the nominal case. By cleverly using component-specific mapping, for a small price, we get fully functioning systems despite extreme variation.

12.6.4 Measurement Precision

VMATCH depends on the ability to characterize the variation in the threshold voltage of the restoring transistors. Taking advantage of the architecture of

Table 12.4 Percent extra energy required for VMATCH to achieve 100% yield on 100 chips at 38% variation as compared to the variation-free case

	% Extra energy			% Extra energy	
Net	Mean	St.dev.	Net	Mean	St.dev.
alu4	14	6	ex5p	13	5
apex2	21	8	frisc	33	15
apex4	13	5	misex3	15	7
bigkey	15	10	pdc	22	5
clma	30	13	s298	19	7
des	19	9	s38417	26	11
diffeq	22	10	s38584.1	29	10
dsip	21	10	seq	16	5
elliptic	20	11	spla	18	5
ex1010	27	9	tseng	21	7

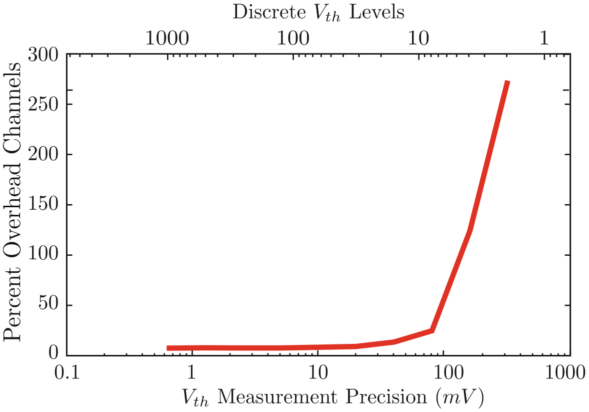
the NanoPLA, voltage dividers can be configured between a restoring nanowire-transistor pair and a reference resistance. Once correctly configured, the voltage divider is used to make a series of measurements. These are used to find the gate to source voltage, V_{gs}, of the restoring transistor that causes the output of the voltage divider to be within a predetermined percent of the source voltage. As fully explained in [35], this V_{gs} directly correlates to the variation of V_{th} in the transistor being measured. Unfortunately, any measurement scheme will have a limited precision over which V_{th}'s variation can be determined. Nevertheless, as long as there are enough resources to choose from, limited precision does not hinder the success of VMATCH.

Though direct isolation of a restore wires within one plane is not possible, it is possible to select one restore wire since diode programing requires the ability to select the restore and output or compute wires connected to the diode being programmed. To prevent interference during measurement, we initially strongly turn all restore transistors off by applying and storing a high V_{gs} voltage so that even the leakiest transistors are strongly disabled. One wire is then selected using the address decoder and V_{gs} for that wire is searched. Using a simple comparator, we find the value of V_{gs} that causes the voltage divider's output to be a predetermined percent (usually 63%) of the source voltage. The precision of this search determines how well we can calculate the variation in the transistor's threshold voltage. The precision of the search depends on the control with which V_{gs} can be changed. As such, we measure precision in terms of ΔV.

Limited precision over V_{gs} has the direct effect of separating the V_{th} variation into a finite number of linearly distributed discrete value ranges. Since R_{offFET} varies exponentially with respect to V_{th} (Equation (12.9)), limited V_{th} precision exponentially distributes the measurement precision of R_{offFET}. Therefore, the leakiest resources, having lower R_{offFET} values, are differentiated, while the slower resources are represented by fewer discrete values. This allows VMATCH to differentiate between a resource that is too leaky and one that is very fast but still acceptable. Therefore, VMATCH can maintain performance even with limited precision.

Figure 12.24 shows the number of extra channels needed to maintain 100% yield as precision decreases. Precision is marked on the lower axis in mV and refers to the smallest precision to which V_{gs} can be adjusted. The upper axis shows the number of unique values of V_{th} the precision produces. From this graph, it is clear that as long as there are enough spare resources, 100% yield can be achieved even at very low precision.

Fig. 12.24 Percent extra channels needed for VMATCH to maintain 100% yield as precision (number of discrete V_{th} levels) decreases. Benchmark spla at $\sigma = 38\%$ 100 Chips



Adding extra resources has a cost, the total delay, energy, and area increases. Figure 12.25 shows how the extra channels needed to maintain 100% yield alter these values as compared to the infinite measurement precision case. It is interesting to note that even when the precision is limited to 40 mV, there is nearly no negative effect on the total delay, energy, and area. This is equivalent to only having 10 discrete values for V_{th} as the upper axis shows. On average over the Toronto 20 Benchmarks, the lowest precision required to achieve result quality within 10% of infinite precision is 45 mV.

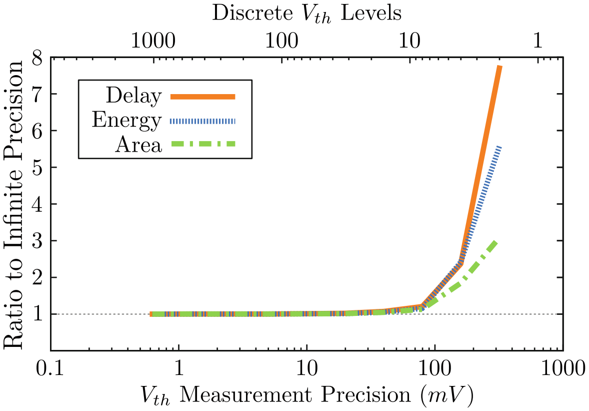


Fig. 12.25 Effect of precision (number of discrete V_{th} levels) on delay, energy and area as a ratio to infinite precision. VMATCH, benchmark spla at $\sigma = 38\%$ 100 chips

12.7 Tolerating Defects

In this section we look at component-specific mapping for defect tolerance. From one perspective, this is the degenerate case of the previous section where we have a single bit of precision – that is, is the resource usable? As we will see, we can actually ask a slightly more focused question, can this resource be used to perform a particular function? We review two examples here, a lightweight component-specific router for CMOS FPGAs based on pre-computed alternatives (Section 12.7.1) and a greedy, matching-based strategy for tolerating non-programmable crosspoint defects in the NanoPLA (Section 12.7.2). These demonstrate the ability to tolerate high defect rates (1% for the FPGA and 5–10% for the NanoPLA). They further demonstrate that testing can be an integral part of mapping and configuration, avoiding the need to generate a complete defect map up front. While originally developed for simple defect tolerance, natural extensions allow these approaches to deal with the more continuous cases that arise with variation and timing.

12.7.1 Pre-computed Alternatives

The need to (1) measure the functionality and performance of individual resources, (2) store large defect maps of the results, (3) perform expensive CAD operations such as placement and routing on a per-component basis, and (4) provide a unique configuration bitstream for each component are significant costs associated with the component-specific mapping approach. Without care, these costs could undermine the yield, performance, and energy benefits of component-specific mapping. In this section, we show how a lightweight, load-time configuration scheme can use a single bitstream to avoid the need to store defect maps and perform expensive CAD. The bitstream is composed of pre-computed alternatives, and the bitstream loader tests alternatives in-system as an integral part of the configuration load process. This scheme was introduced in [72].

12.7.1.1 Idea

The basic idea is that we pre-determine multiple ways to satisfy each logical function required by the logical netlist. For a net connecting, two logic blocks, we identify multiple paths using different resources through the interconnect. For logic, we identify multiple compute blocks where the logic can be performed. Instead of just storing one mapping in the bitstream for each logical function, we store all of these alternatives. This defers the decision of which resource to use until configuration load time. At load time, the bitstream loader simply needs to select one of the alternatives that both (a) avoids resources already in use and (b) avoids unusable resources. In fact, once it has established the resources required for the alternative are not in use, it can configure the function and test it out. The loader does not actually need to know the functionality of each of the constituent resources; it

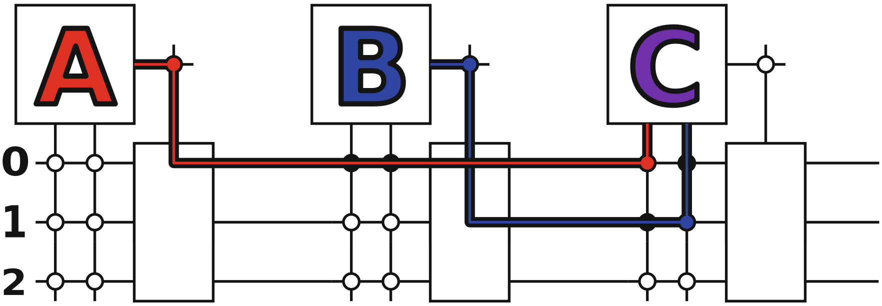


Fig. 12.26 Example route configuration without defects

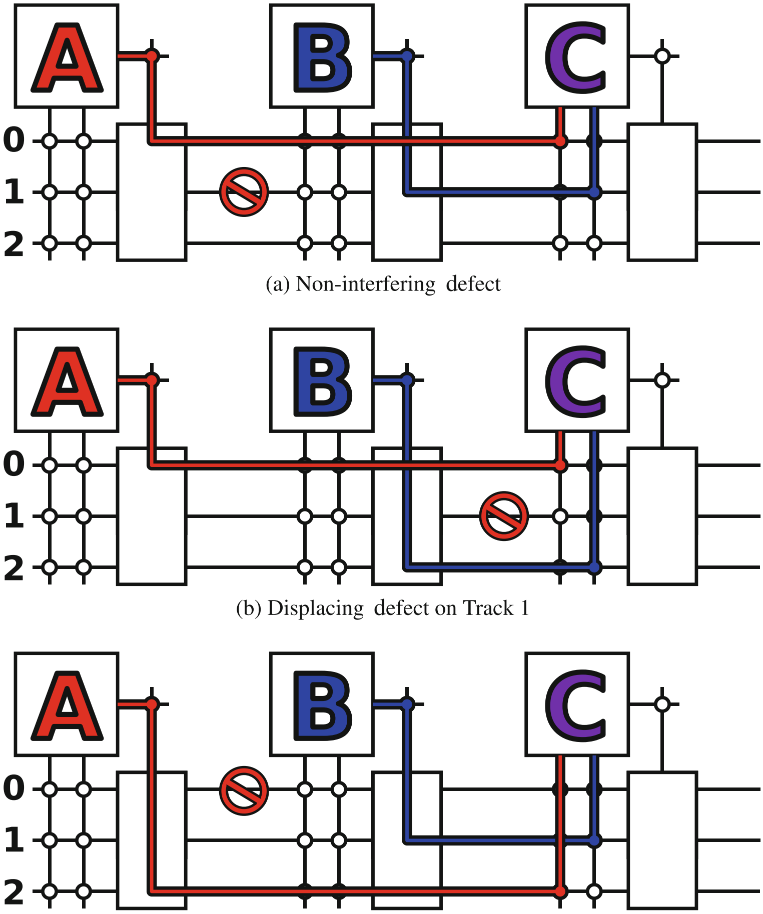
can perform an end-to-end test and see if the configured resource set provides the requisite functionality.

To illustrate this idea, we revisit our simple FPGA example from Section 12.2 as shown in Fig. 12.26. Here, we need to provide logical connections from A→C and B→C. When we compute the routing, rather than simply storing the single configuration suggested for the connections from A→C and B→C, we store alternative routes for each of the logical connections. For example, we might note that we could route A→C on either track 0 or track 2, and we could route B→C on either track 1 or track 2. At load time, the loader tests each of the alternatives, perhaps using the scan techniques from Section 12.3.1, until it finds one that will work. If, as shown in Fig. 12.27a, the only defect occurs on an unused segment, the initial configuration with A→C on track 0 and B→C on track 1 is acceptable. However, if a defect on track 0 or 1 interferes with these routes, the alternative route gives the loader an option to avoid the defect as shown in Fig. 12.27b, c.

We can similarly pre-compute alternatives for logic. The simplest strategy might be to allocate a spare LUT within each clustered logic block and use the spare as a local alternative to each LUT [51]. A more general strategy allocates local spares periodically (e.g., in every $k \times k$ region) and includes alternative placements that avoid any small subset of errors within that region [50] (see Fig. 12.28). This can be used in architectures that do not use clusters or as a hierarchical alternative strategy when local substitution within a cluster fails.

12.7.1.2 Pre-computed Alternative Generation Strategy

Our goal is to keep the load-time complexity low both to assure that the hardware support is small and to minimize the time required for the load. We aim for a greedy load-time algorithm that can sequentially evaluate each set of resources. To satisfy this goal without sacrificing route quality, we adopt a hybrid strategy for generating alternatives that includes (1) a base assignment along with (2) shared alternatives. That is, we start by identifying a reserved set of resources for spares (e.g., a set of route tracks and logic blocks). We then place and route the design normally on the



(a) Non-interfering defect

(b) Displacing defect on Track 1

(c) Displacing defect on Track 0

Fig. 12.27 Example using pre-computed alternatives to avoid defects

non-reserved set of resources. After the base assignment is completed, we then identify alternate ways to satisfy each logical function (e.g., each logical net and LUT) from the reserved set of spare resources. While we demand that each function have a unique resource in the base mapping, we allow functions to share resources for the alternatives. For low defect rates, most resources are satisfied with their original assignment in the base set. Only a small fraction is displaced, and only this fraction contends for the reserved spares during load time.

Returning to our route example above (Figs. 12.26 and 12.27), we reserve Track 2 and perform the base route using Tracks 0 and 1 as shown in Fig. 12.26. Then we generate an alternative route using Track 2 for each of the nets (A→C and B→C). Alternatives are ordered in the bitstream with the base route first. If the base route

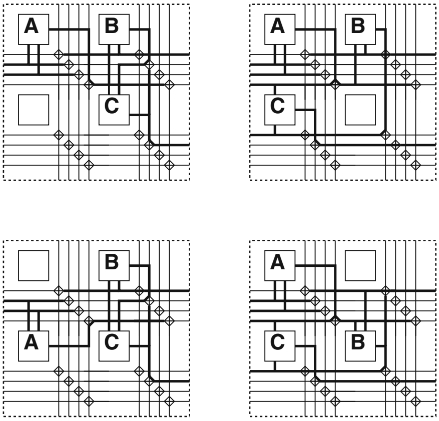


Fig. 12.28 Four placements of three gate subgraph on a 2×2 tile

is functional, it is used (Fig. 12.27a). When the base route is non-functional, it tries the alternative (Fig. 12.27b, c).

12.7.1.3 Impact

Figure 12.29 shows the result of these precomputed alternatives for the `des` benchmarks from the Toronto 20 benchmark set [10]. For the case shown, we perform a low-stress route allowing the base route to use 20% extra channels over the minimum number of routable channels, and we reserve an additional 20% channels as spare tracks for alternatives. For this experiment, interconnect switches and wires fail randomly and independently with the probability shown on the X-axis. Data at each defect rate is based on a collection of 100 chips each with random defects generated according to the associated defect rate. The "Perfect" case corresponds to the probability that a component large enough to hold this benchmark is defect

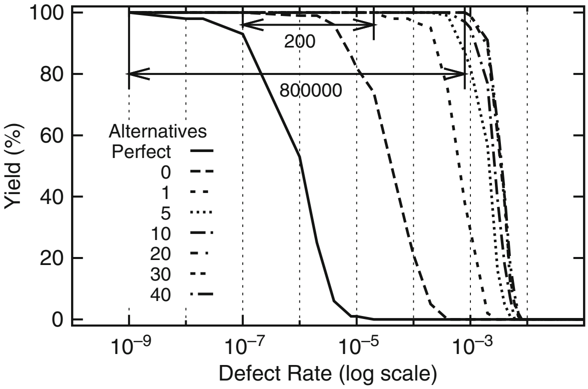


Fig. 12.29 Yield vs. defect rate for `des`

free – this is the yield rate at which an FPGA vendor can sell a component to be used with any design using a conventional, component-oblivious approach. The "0 Alternatives" case would correspond to the Xilinx EasyPath case (Section 12.4). This is the probability that a manufactured chip works on the base route mapping. This yield rate is higher than the "Perfect" case since the base route mapping does not use all the resources on the FPGA. The 1–40 "Alternatives" curves show the achievable yield rate when using the associated number of alternatives. Increasing the number of alternatives increases the yield rate, but there are diminishing benefits as the alternatives begin to approach the number of distinct, alternative paths in the underlying component.

The ratio annotations (200 and 800,000) are marked where the yield drops form 100% yield according to our experiment. The first annotation highlights that a single alternative allows the component to achieve near 100% yield for a factor of 200 larger defect rate than the component-specific (0 Alternative or EasyPath) case. The second annotation highlights a factor of 800,000 increase in tolerable defect rate from the perfect yield case to the 40 alternative case. At 40 alternatives, this scheme can tolerate roughly a 1% defect rate of switches and wires while still achieving near 100% yield.

12.7.1.4 Extensions

The basic technique of pre-computed alternatives can also be used for variation. As noted above, by including timing requirements and substituting a timing test for the correctness test, alternatives can be used to identify and replace slow paths on a fabricated component. Building on the performance and functionality test, pre-computed alternatives can also be used to minimize energy. That is, by performing the load operation at a specific supply voltages, we are asking the loader to try to find a set of alternatives that is acceptable at that voltage. When the voltage becomes too low, the loader will not be able to find adequate resources to repair the design and successfully complete the bitstream load process. With this core capability, an outer loop can perform a binary search on the supply voltage to determine the smallest operating voltage the component will allow. Note that the loader still only asks a binary question on each path test: is this alternative currently good enough to provide the functionality required?

12.7.2 NanoPLA Crosspoint Matching

In this section we deal with the challenge of non-programmable crosspoints in crossbar architectures. When the defect rates are in the 1–10% range, standard approaches like row and column sparing would require orders of magnitude overhead. In contrast, we show that component-specific methods can tolerate these defects with little or no overhead. Similar to VMATCH (Section 12.6.3), the trick is to match the requirements of each function to the capabilities of each physical NAND term. This allows us to exploit resources that a conservative, design-oblivious mapping would discard. The scheme described here was introduced in [30, 63]. It is similar in spirit to approaches developed in [24, 80, 84].

12.7.2.1 Challenge: Non-programmable Crosspoints

In the NanoPLA (Section 12.5.2) and similar crossbar architectures (e.g., [38, 59, 71, 80, 84]) programmable crosspoints are just a few nanometers wide, meaning the entire cross-sectional area of the crosspoint may be in the order of 10 square nanometers. As a result, proper function of the crosspoint may depend on a small number of atomic or molecular bonds whose connections are statistical in nature. In cases using molecular films (e.g., [15, 17, 20]), some film gridpoints may not be filled with molecules, leading to crosspoints that contain fewer or no molecules. This suggests a common failure mechanism is *non-programmable crosspoints* that cannot be programmed into a sufficiently low-resistance "on" state.

In the earliest demonstrations of the molecular switch crossbar [17], researchers reported that 85% of the junctions were functional. A denser crossbar [40] showed even higher defect rates and provided evidence supporting the random distribution of defects in the crossbar array. We expect crosspoint yield to improve as these processes are further optimized and move from the research lab to manufacturing. Nonetheless, devices constructed from a small number of statistically assembled bonds will remain prone to relatively high defect rates.

12.7.2.2 Conventional, Design-Oblivious Approaches

For the sake of illustration, consider a non-programmable crosspoint failure rate of 5%, or a yield rate of $P_{\text{xpt}} = 0.95$. A typical NanoPLA NAND-term (Section 12.6.3.1) is composed of 100 crosspoints. The probability that a NanoPLA NAND-term has a full set of 100 good crosspoints is

$$P_{\text{perfect\_nand}} = (0.95)^{100} \approx 0.0059 \tag{12.10}$$

That is, we have less than a 0.6% chance of yielding a perfect NAND-term.

A design-oblivious approach might use row-sparing as is commonly used in memories. We could add spare rows (NAND-terms) and avoid the imperfect rows. This way we can map any design to the remaining perfect rows. When the probability of yielding an entire row is high, row sparing works well. If we had $P_{\text{perfect\_nand}} > 0.999$, then a few spares, constituting just a few percent area overhead, would be adequate to repair the array and allow us to obliviously map any design to the set of perfect NAND-terms. Unfortunately, with $P_{\text{perfect\_nand}} < 0.006$, it would require tens of thousands of (NAND-terms) to yield 100 perfect (NAND-terms), suggesting at least two orders of magnitude area overhead.

12.7.2.3 Component-Specific Observations

The first important component-specific observations is that a particular logical NAND in the design only needs a subset of the junctions in a physical NAND-term to be programmable. That is, it does not require a perfect NAND-term, but rather one that is "good enough" for the particular task.

Consider the small NAND-term array and defect pattern shown in Fig. 12.30a. Now consider the function $F_5 = \overline{(A \cdot B \cdot D)}$. F_5 can use NAND-term N_1 despite the fact that the NAND-term has a defective crosspoint (C, N_1). The logical function F_5 does not need the functionality provided by that crosspoint. In general, F_5 can use any NAND-term with programmable crosspoints for the three required junctions (A, B, and D). The probability that any NAND-term can support F_5 is

$$P_{\text{good\_enough\_for}}(F_5) = \left(P_{xpt}\right)^3 \tag{12.11}$$

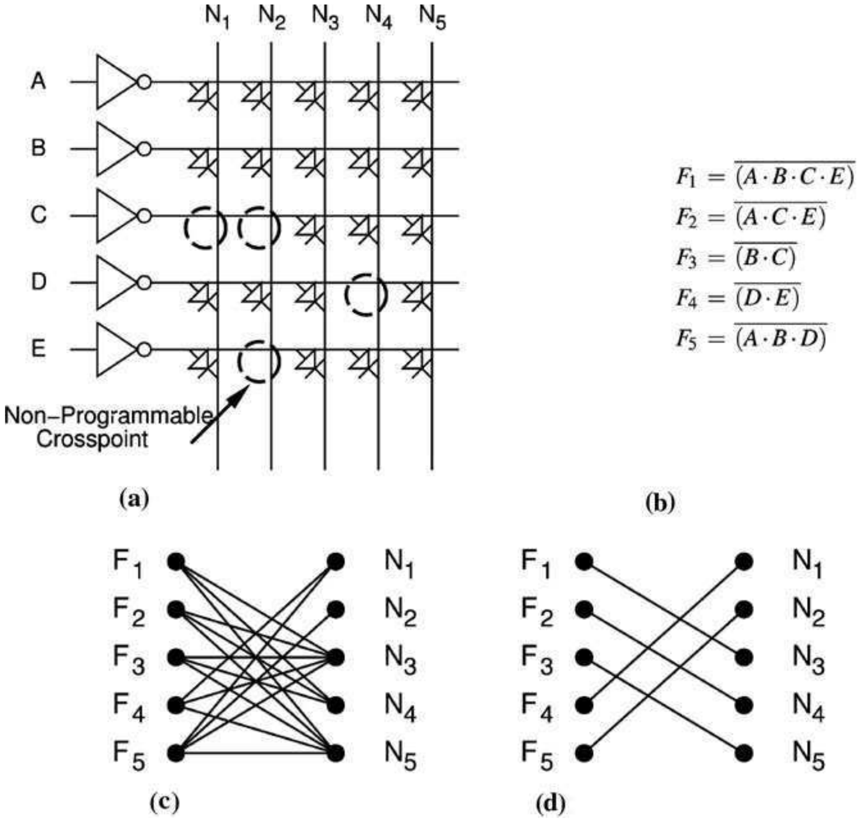

N1 N2 N3 N4 N5
A
B
C
D
E
Non–Programmable Crosspoint
(a)
$F_1 = \overline{(A \cdot B \cdot C \cdot E)}$
$F_2 = \overline{(A \cdot C \cdot E)}$
$F_3 = \overline{(B \cdot C)}$
$F_4 = \overline{(D \cdot E)}$
$F_5 = \overline{(A \cdot B \cdot D)}$
(b)
F1 F2 F3 F4 F5
N1 N2 N3 N4 N5
(c)
F1 F2 F3 F4 F5
N1 N2 N3 N4 N5
(d)

Fig. 12.30 Defective crossbar example. (**a**) Crosspoint defects; (**b**) Logical NAND-terms; (**c**) Bipartite matching; (**d**) NAND-term assignment

With $P_{\mathrm{xpt}} = 0.95$, we have $P_{\mathrm{good\_enough\_for}}(F_5) = 0.857$, which is much higher than $P_{\mathrm{perfect\_nand}}$. Nonetheless, a NAND-term yield rate of 85% could still demand high overhead.

The second important component-specific observations is that an array of NAND-terms contains many NAND-terms to choose from. We do not have to use a particular NAND-term, but can search through the array to find one that can support our logic.

Returning to our example in Fig. 12.30, $F_5 = \overline{(A \cdot B \cdot D)}$ cannot use NAND-term N_4 since it has a defective (D, N_4) crosspoint, but it can, in fact, use any of the remaining NAND-terms shown. In general, the probability that we can find a nanowire that is good enough in a collection of N NAND-terms is

$$P_{\mathrm{one\_is\_good\_enough\_for}}(F) = 1 - \left(1 - P_{\mathrm{good\_enough\_for}}(F)\right)^N \tag{12.12}$$

For $P_{\mathrm{good\_enough\_for}}(F_5) = 0.857$ and $N = 5$, we see that the probability that there is one NAND-term good enough for F_5 is over 0.9999. This suggests we may need very few spares, if any, to achieve a complete mapping.

12.7.2.4 Matching

In practice, we need to find compatible assignments for all of the logical NAND-terms in a NanoPLA plane. That is, if there is only a single physical NAND-term that is the only NAND-term good enough for multiple logical NAND-terms we will not be able to provide a unique NAND-term for each function. Fortunately, with arrays of 100 NAND-terms, there is a high probability that there will be multiple physical NAND-terms that can implement each logical function.

We can formulate this problem formally as a bipartite matching problem. The matching is an assignment of a unique physical NAND-term for each logical NAND-term. Starting with the set of logical and physical NAND-terms as our two sets, we add links between each logical NAND-term and each of the physical NAND-terms that supports it (See Fig. 12.30c). Then we look for a suitable assignment. This can be done optimally using polynomial time network flow algorithms [3, 22, 46]. Alternately, a greedy heuristic that assigns the first suitable NAND-term to a function without backtracking runs in linear time and is adequate for typical NanoPLA mapping tasks [63].

12.7.2.5 Results

Table 12.5 shows the resulting area impact when mapping the Toronto 20 [10] benchmark set to NanoPLAs with various defect rates [30]. Overhead comes from the need to allocate spare NAND-terms to assure successful mappings and from area required to decompose high fanin functions (reduce the number of good programmable functions required per logical NAND-term) to ease mapping at higher defect rates. At 5% non-programmable crosspoint defect rates, all designs can be mapped without any additional spares. Note that this is in stark contrast to the oblivious, row-sparing-based mapping (Section 12.7.2.2) that required at least two

Table 12.5 Relative area versus P_{xpoint} (nanowire pitch is 10 nm; reliable, lithographic support pitch is 105 nm)

	P_{xpt}			
Design	0.85	0.9	0.95	1.00
alu4	1.81	1.64	1.00	1.00
apex2	1.19	1.19	1.00	1.00
apex4	1.30	1.16	1.00	1.00
bigkey	1.00	1.00	1.00	1.00
clma	1.00	1.00	1.00	1.00
des	1.00	1.00	1.00	1.00
dsip	1.00	1.00	1.00	1.00
elliptic	1.00	1.00	1.00	1.00
ex1010	3.81	2.15	1.00	1.00
ex5p	1.00	1.00	1.00	1.00
frisc	1.00	1.00	1.00	1.00
misex3	1.31	1.31	1.00	1.00
pdc	4.75	1.79	1.00	1.00
s298	1.84	1.84	1.00	1.00
seq	1.20	1.12	1.00	1.00
spla	3.46	1.83	1.00	1.00

orders of magnitude overhead for sparing. Some designs continue to find successful mappings with no spares even at 15% defect rates while others show increasing overhead.

The component-specific approach allows us to use almost all the NAND *terms despite the fact that almost every nanowire is imperfect.*

12.7.2.6 Timing Extension

The results from [30] simulate binary failures – a junction is either usable or unusable. However, we can reasonably expect a variation in the "on" resistance of the crosspoints [94] as well as the nanowire resistances. As such, we might change the definition of "good enough" to include a performance require – does this physical NAND-term support this logical NAND-term within a target plane evaluation time period? This may require a series of timing tests to establish compatibility but otherwise admits to the same matching formulation.

12.8 Lifetime Repair of In-Field Failures

Once the design is prepared to perform self-diagnosis and repair, it is also possible to address aging (Chapter 1, Section 1.4) and hard errors that arise during operation. For example, we can "reboot" the component to force it to re-execute a load-time defect and variation tolerance scheme such as the one described in Section 12.7.1. The bitstream loader will test and avoid all the new, persistent defects and resources that are now too slow (or leaky) to perform their previously assigned roles just as it does those defects that resulted from manufacture.

12.8.1 System and Technique Requirements

In order to perform in-field repairs, the deployed system platform must be capable of running both the resource characterization and the component-specific remapping algorithm. The pre-computed alternative algorithm described above (Section 12.7.1) was designed to integrate characterization with configuration, demand minimum resources to support configuration, and perform the component-specific mapping relatively quickly. A complete NanoPLA chip would embed a small, reliable CMOS configuration processor on the die [30] to perform algorithms like VMATCH (Section 12.6.3) and greedy crosspoint matching (Section 12.7.2). Both of these algorithms run in linear-time, are designed for on-chip measurements, and operate on minimum state (one PLA at a time).

Fully general place-and-route algorithms typically require large memory footprints and fairly powerful processors. These resources may not be available on all embedded system platforms. Steiner illustrates how the PowerPC processor on a Virtex-II Pro [95] can be used to perform in-system placement and routing and allow defect avoidance [82, 83].

12.8.2 Incremental Repair

Starting from scratch and re-characterizing and re-mapping the entire component is the most general solution. It fully reuses techniques that will already be necessary to accommodate fabrication variation. However, it can be time-consuming, potentially leaving the system out of service for a long period of time while it runs the algorithm. Even with fast, in-system techniques this can be milliseconds to seconds [72] or minutes [82].

If we can localize the new error, it should be possible to incrementally repair the component in significantly less time than a full characterization and mapping. For example, the pre-computed alternative case (Section 12.7.1) could simply reload the alternatives for a single logical block or two-point connection. Similarly, the NanoPLA schemes could locally remap a single PLA block rather than the entire array.

12.8.3 Detection of In-Field Failures

Detection is a key issue for in-field failures. How do we know when something has failed and we need to perform repair?

12.8.3.1 Logic Scrubbing

The most straightforward solution is to perform periodic, interleaved testing (e.g., [33, 78]). That is, at some regular intervals, we stimulate the system with a set of vectors with known outputs or signatures. In the extreme, this is a periodic complete, design-specific test of the logic. If the component passes the self test, we know that it is still working and the data produced since the last such self test was not corrupted by aging or persistent errors. When the component fails, we know that there is a need for repair. Furthermore, we know the data since the last test is suspect. In some systems, that may demand a rollback of system state and recomputation of the suspect results (e.g., [4]). With careful design, these periodic tests can localize the impact of the new errors, helping facilitate incremental repair (Section 12.8.2).

These periodic test and repair operations are similar to scrubbing in memories (e.g., [74]). That is, it is easiest and cheapest to detect errors when only one or a few show up a time. Similarly, when only a few errors occur it is easier to localize them. In the memory ECC case, repair is often guaranteed if only a single error has occurred in each memory word. Periodic scrubbing of the logic or the memory increases the likelihood that we will find errors before multiple errors can accumulate and confound detection, repair, and localization [27].

12.8.3.2 Self-Monitoring

It may also be possible to detect persistent errors and aging by monitoring the normal processing of application data. This may accelerate detection of errors, both allowing more timely repair and reducing the state that must be kept to recover from errors.

Lightweight techniques exist to detect a class of timing errors that may arise from aging or environmental changes [9, 14]. Specifically, by sampling the inputs to registers slightly after the clock, it is possible to identify late arriving signal transitions. These late signals indicate when the system clock is exceeding the capabilities of the circuit and can be used for in-system voltage-scaling or clock frequency tuning. They can also serve as a trigger for repair operations.

The misbehavior of software running on top of the component may also serve as an important early warning that can trigger diagnostic and repair routines. In particular, persistent failures can lead to violation of known software invariants or atypical software behavior [56, 73]. Checking the software invariants and noting unexpected program misbehavior (e.g., excessive loop counts, traps, illegal instruction and data references, excessive OS activity) catches many instances of hard failures in processors.

12.9 Trends and Future Prospects

As the examples in this chapter demonstrate, component-specific mapping allows us to tolerate very high defect rates, extreme variation, and lifetime aging. The component-specific model forces us to change from a one-mapping-fits-all model to a per-component, in-system mapping model. Nonetheless, the early demonstrations illustrated here suggest that the benefits are compelling and the new challenges are manageable.

Continued feature-size scaling and the move to post-CMOS technologies all suggest increasing variation, defect rates, and lifetime aging effects. The level of defects and variation will exceed our conventional solutions for fixed-function components and oblivious mappings for programmables. This suggests a growing advantage for FPGA-like components exploiting component-specific techniques and an opportunity to further narrow or close the traditional power gap between FPGAs and custom ASICs (Chapter 10, Section 10.2.2). Fixed-function components are already adopting the most coarse-grained techniques for component adaptation and defect tolerance (e.g., adaptive biasing, post-silicon clock tuning (Section 12.4), core sparing). As defect and variation rates increase, these components will be driven to increasing use of post-fabrication configuration (e.g., [8]). That is, *all components must become configurable and adaptable.* The questions moving forward will not be "can we afford to use configurable components," but "how configurable do our components need to be?" That is, we must carefully consider how we tradeoff energy with flexibility while providing the appropriate level of configurability to accomodate the expected rates of defects, variations, and aging.

References

1. ITRS (2008) International technology roadmap for semiconductors. <http://www.itrs.net/Links/2008ITRS/Home2008.htm>
2. Luu J, Jamieson P, Kuon I, Betz V, Marquardat A, Rose J (2008) VPR and T-VPack: versatile Packing, Placement and Routing for FPGAs. <http://www.eecg.utoronto.ca/vpr/>
3. Alt H, Blum N, Mehlhorn K, Paul M (1991) Computing a maximum cardinality matching in a bipartite graph in time o(n1.5 pm/log (n)). Inf Process Lett 37(4):237–240. doi: http://dx.doi.org/10.1016/0020-0190(91)90195-N
4. Asadi GH, Tahoori MB (2005) Soft error mitigation for SRAM-based FPGAs. In: Proceedings of the VLSI Test Symposium, pp 207–212
5. Asenov A (1998) Random dopant induced threshold voltage lowering and fluctuations in sub- 0.1 μm MOSFET's: A 3-D "atomistic" simulation study. IEEE Trans Electron Devices 45(12):2505–2513
6. Asenov A (2002) Intrinsic threshold voltage fluctuations in decanano MOSFETs due to local oxide thickness variation. IEEE Trans Electron Devices 49(1):112–119
7. Asenov A, Kaya S, Brown AR (2003) Intrinsic parameter fluctuations in decananometer MOSFETs introduced by gate line edge roughness. IEEE Trans Electron Devices 50(5):1254–1260
8. Ashoue M, Chatterjee A, Singh DA (2010) Post-manufacture tuning for Nano-CMOS yield recovery using reconfigurable logic. IEEE Trans VLSI Syst 18(4):675–679. doi: 10.1109/TVLSI.2009.2014559
9. Austin T, Blaauw D, Mudge T, Flautner K (2004) Making typical silicon matter with Razor. IEEE Compu 37(3):57–65
10. Betz V, Rose J (1999) FPGA place-and-route challenge. <http://www.eecg.toronto.edu/vaughn/challenge/challenge.html>

11. Bijansky S, Aziz A (2008) TuneFPGA: post-silicon tuning of dual-Vdd FPGAs. In: Proceedings of the ACM/IEEE Design Automation Conference
12. Boning D, Panganiban J, Gonzalez-Valentin K, Nassif S, McDowell C, Gattiker A, Liu F (2002) Test structures for delay variability. In: Proceedings of the international workshop on timing issues in the specification and synthesis of digital systems. ACM, New York, NY, pp 109
13. Borghetti J, Snider GS, Kuekes PJ, Yang JJ, Stewart DR, Williams RS (2010) 'Memristive' switches enable 'stateful' logic operations via material implication. Nature 464(7290): 873–876. doi: 10.1038/nature08940. URL http://dx.doi.org/10.1038/nature08940
14. Bowman KA, Tschanz JW, Kim NS, Lee JC, Wilkerson CB, Lu SLL, Karnik T, De VK (2009) Energy-efficient and metastability-immune resilient circuits for dynamic variation tolerance. IEEE J Solid State Circuits 44(1):49–63
15. Brown CL, Jonas U, Preece JA, Ringsdorf H, Seitz M, Stoddart JF (2000) Introduction of [2]catenanes into langmuir films and langmuir-blodgett multilayers. a possible strategy for molecular information storage materials. Langmuir 16(4): 1924–1930
16. Cao Y, Clark L (2007) Mapping statistical process variations toward circuit performance variability: an analytical modeling approach. IEEE Trans Comput-Aided Des Integr Circ Syst 26(10):1866–1873
17. Chen Y, Jung GY, Ohlberg DAA, Li X, Stewart DR, Jeppesen JO, Nielsen KA, Stoddart JF, Williams RS (2003) Nanoscale molecular-switch crossbar circuits. Nanotechnology 14: 462–468
18. Chen Y, Ohlberg DAA, Li X, Stewart DR, Williams RS, Jeppesen JO, Nielsen KA, Stoddart JF, Olynick DL, Anderson E (2003) Nanoscale molecular-switch devices fabricated by imprint lithography. Appl Phys Lett 82(10):1610–1612
19. Cheng L, Xiong J, He L, Hutton M (2006) FPGA performance optimization via chipwise placement considering process variations. In: Proceedings of the international conference on field-programmable logic and applications, pp 1–6
20. Collier C, Mattersteig G, Wong E, Luo Y, Beverly K, Sampaio J, Raymo F, Stoddart J, Heath J (2000) A [2]Catenane-Based Solid State Reconfigurable Switch. Science 289: 1172–1175
21. Committee I.S. (1990) IEEE standard test access port and boundary-scan architecture. IEEE, 345 East 47th Street, New York, NY 10017-2394. IEEE Std 1149.1-1990
22. Cormen T, Leiserson C, Rivest R (1990) Introduction to Algorithms. MIT Press and McGraw, Cambridge, Massachusetts, New York.
23. Cui Y, Lauhon LJ, Gudiksen MS, Wang J, Lieber CM (2001) Diameter-controlled synthesis of single crystal silicon nanowires. Appled Phys Lett 78(15):2214–2216
24. Culbertson WB, Amerson R, Carter R, Kuekes P, Snider G (1997) Defect tolerance on the TERAMAC custom computer. In: Proceedings of the IEEE symposium on FPGAs for custom computing machines, pp 116–123
25. DeHon A (2005) Design of programmable interconnect for sublithographic programmable logic arrays. in: proceedings of the international symposium on field-programmable gate arrays, pp 127–137
26. DeHon A (2005) Nanowire-based programmable architectures. ACM J Emerg Technol Comput Syst 1(2):109–162. doi: http://doi.acm.org/10.1145/1084748.1084750
27. DeHon A (2008) The case for reconfigurable components with logic scrubbing: regular hygiene keeps logic FIT (low). In: Proceedings of the international workshop on design and test of nano devices, circuits, and systems, pp 67–70
28. DeHon A, Likharev KK (2005) Hybrid CMOS/nanoelectronic digital circuits: Devices, architectures, and design automation. In: Proceedings of the international conference on computer aided design, pp 375–382
29. DeHon A, Lincoln P, Savage J (2003) Stochastic assembly of sublithographic nanoscale interfaces. IEEE Trans Nanotechnol 2(3):165–174

30. DeHon A, Naeimi H (2005) Seven strategies for tolerating highly defective fabrication. IEEE Des Test Comput 22(4):306–315
31. DeHon A, Wilson MJ (2004) Nanowire-based sublithographic programmable logic arrays. In: Proceedings of the international symposium on field-programmable gate arrays, pp 123–132
32. Dong Y, Yu G, McAlpine MC, Lu W, Lieber CM (2008) Si/a-Si core/shell nanowires as nonvolatile crossbar switches. Nanoletters 8(2):386–391
33. Emmert J, Stroud C, Skaggs B, Abramovici M (2000) Dynamic fault tolerance in FPGAs via partial reconfiguration. In: Proceedings of the IEEE symposium on field-programmable custom computing machines, pp 165–174
34. Fan Z, Mo X, Lou C, Yao Y, Wang D, Chen G, Lu JG (2005) Structures and electrical properties for Ag-tetracyanoquinodimetheane organometallic nanowires. IEEE Trans Nanotechnol 4(2):238–241
35. Gojman B (2010) Algorithms and techniques for conquering extreme physical variation in bottomup nanoscale systems. Master's thesis, California Institute of Technology. http://resolver.caltech.edu/CaltechTHESIS:04052010-152122284
36. Gojman B, DeHon A (2009) VMATCH: using logical variation to counteract physical variation in bottom-up, nanoscale systems. In: Proceedings of the international conference on field-programmable technology, pp 78–87. IEEE
37. Gojman B, Manem H, Rose GS, DeHon A (2009) Inversion schemes for sublithographic programmable logic arrays. IET Comput Digit Tech 3(6):625–642
38. Goldstein SC, Budiu M (2001) NanoFabrics: spatial computing using molecular electronics. In: Proceedings of the international symposium on computer architecture, pp 178–189
39. Goldstein SC, Rosewater D (2002) Digital logic using molecular electronics. In: ISSCC digest of technical papers, pp 204–205. IEEE
40. Green JE, Choi JW, Boukai A, Bunimovich Y, Johnston-Halperin E, DeIonno E, Luo Y, Sheriff BA, Xu K, Shin YS, Tseng HR, Stoddart JF, Heath JR (2007) A 160- kilobit molecular electronic memory patterned at 10^{11} bits per square centimetre. Nature 445:414–417
41. Gudiksen MS, Lauhon LJ, Wang J, Smith DC, Lieber CM (2002) Growth of nanowire superlattice structures for nanoscale photonics and electronics. Nature 415: 617–620
42. Gudiksen MS, Wang J, Lieber CM (2001) Synthetic control of the diameter and length of semiconductor nanowires. J Phys Chem B 105:4062–4064
43. Hanson S, Zhai B, Bernstein K, Blaauw D, Bryant A, Chang L, Das KK, Haensch W, Nowak EJ, Sylvester DM (2006) Ultralow-voltage, minimum-energy CMOS. IBM J Res Dev 50(4–5):469–490
44. Hauck S, DeHon A (eds) (2008) Reconfigurable computing: the theory and practice of FPGA based computation. Systems-on-Silicon. Elsevier, Burlington, MA
45. Heath JR, Kuekes PJ, Snider GS, Williams RS (1998) A defect-tolerant computer architecture: opportunities for nanotechnology. Science 280(5370):1716–1721
46. Hopcroft JE, Karp RM (1973) An n2.5 algorithm for maximum matching in bipartite graphs. SIAM J Comput 2(4):225–231
47. Huang Y, Duan X, Wei Q, Lieber CM (2001) Directed assembly of one-dimensional nanostructures into functional networks. Science 291:630–633
48. Katsuki K, Kotani M, Kobayashi K, Onodera H (2005) A yield and speed enhancement scheme under within-die variations on 90 nm LUT array. In: Proceedings of the IEEE custom integrated circuits conference, pp 601–604
49. Krishnan G (2005) Flexibility with EasyPath FPGAs. Xcell J 55:96–98
50. Lach J, Mangione-Smith WH, Potkonjak M (1998) Low overhead fault-tolerant FPGA systems. IEEE Trans VLSI Syst 26(2):212–221
51. Lakamraju V, Tessier R (2000) Tolerating operational faults in cluster-based FPGAs. In: Proceedings of the international symposium on field-programmable gate arrays, pp 187–194

52. Lauhon LJ, Gudiksen MS, Wang D, Lieber CM (2002) Epitaxial core-shell and core-multishell nanowire heterostructures. Nature 420:57–61
53. Law M, Goldberger J, Yang P (2004) Semiconductor nanowires and nanotubes. Annu Rev Mater Sci 34:83–122
54. Lewis D, Ahmed E, Baeckler G, Betz V, Bourgeault M, Cashman D, Galloway D, Hutton M, Lane C, Lee A, Leventis P, Marquardt S, McClintock C, Padalia K, Pedersen B, Powell G, Ratchev B, Reddy S, Schleicher J, Stevens K, Yuan R, Cliff R, Rose J (2005) The Stratix-II logic and routing architecture. In: Proceedings of the international symposium on field-programmable gate arrays, pp 14–20
55. Lewis D, Betz V, Jefferson D, Lee A, Lane C, Leventis P, Marquardt S, McClintock C, Pedersen B, Powell G, Reddy S, Wysocki C, Cliff R, Rose J (2003) The Stratix routing and logic architecture. In: Proceedings of the international symposium on field- programmable gate arrays, pp 12–20
56. Li ML, Ramachandran P, Sahoo SK, Adve SV, Adve VS, Zhou Y (2008) Understanding the propagation of hard errors to software and implications for resilient system design. In: Proceedings of the international conference on architectural support for programming languages and operating systems, pp 265–276
57. Lin X, Press R, Rajski J, Reuter P, Rinderknecht T, Swanson B, Tamarapalli N (2003) High-frequency, at-speed scan testing. IEEE Des Test Comput 20(5):17–25
58. Ling ZM, Cho J, Wells RW, Johnson CS, Davis SG (2003) Method of using partially defective programmable logic devices. US Patent 6,664,808
59. Luo Y, Collier P, Jeppesen JO, Nielsen KA, Delonno E, Ho G, Perkins J, Tseng HR, Yamamoto T, Stoddart JF, Heath JR (2002) Two-dimensional molecular electronics circuits. ChemPhysChem 3(6):519–525
60. Matsumoto Y, Hioki M, Koike TKH, Tsutsumi T, Nakagawa T, Sekigawa T (2008) Suppression of intrinsic delay variation in FPGAs using multiple configurations. Trans Reconfig Technol Syst 1(1). Doi: http://doi.acm.org/10.1145/1331897.1331899
61. McMurchie L, Ebeling C (1995) PathFinder: a negotiation-based performance-driven router for FPGAs. In: Proceedings of the international symposium on field-programmable gate arrays, ACM, pp 111–117
62. Nabaaz G, Aziziy N, Najm FN (2006) An adaptive FPGA architecture with process variation compensation and reduced leakage. In: Proceedings of the ACM/IEEE design automation conference, pp 624–629
63. Naeimi H, DeHon A (2004) A greedy algorithm for tolerating defective crosspoints in NanoPLA design. In: Proceedings of the international conference on field-programmable technology, IEEE pp 49–56
64. Nagaraj K, Kundu S (2009) Process variation mitigation via post silicon clock tuning. In: Proceedings of the Great Lakes symposium on VLSI, pp 227–232
65. Parker KP (1992) The boundary-scan handbook. Kluwer, Norwell, MA
66. Paul S, Bhunia S (2008) MBARC: a scalable memory based reconfigurable computing framework for nanoscale devices. In: Proceedings of the Asia and South Pacific design automation conference, pp 77–82
67. Paul S, Chatterjee S, Mukhopadhyay S, Bhunia S (2009) Nanoscale reconfigurable computing using non-volatile 2-d sttram array. In: Proceedings fo the IEEE international conference on nanotechnology
68. Peng SF (1996) Method and apparatus for testing semiconductor devices at speed. US Patent 5,524,114
69. Rabaey JM, Chandrakasan A, Nikolic B (1999) Digital integrated circuits, 2nd edn. Prentice Hall, Upper Saddle River, New Jersey
70. Radovanovic PV, Barrelet CJ, Gradecak S, Qian F, Lieber CM (2005) General syntehsis of manganese-doped II-VI and III-V semiconductor nanowires. Nanoletters 5(7): 1407–1411

71. Rose GS, Stan MR (2007) A programmable majority logic array using molecular scale electronics. IEEE Trans Circuits Syst I Fundam Theory Appl 54(11): 2380–2390
72. Rubin R, DeHon A (2009) Choose-your-own-adventure routing: lightweight load-time defect avoidance. In: Proceedings of the international symposium on field-programmable gate arrays, pp 23–32
73. Sahoo SK, Li ML, Ramachandran P, Adve SV, Adve VS, Zhou Y (2008) Using likely program invariants to detect hardware errors. In: Proceedings of the international conference on dependable systems and networks, pp 70–79
74. Saleh A, Serrano J, Patel J (1990) Reliability of scrubbing recovery-techniques for memory systems. IEEE Trans Reliab 39(1):114–122
75. Saxena J, Butler KM, Gatt J, Raghuraman R, Kumar SP, Basu S, Campbell DJ, Berech J (2002) Scan-based transition fault testing – implementation and low cost test challenges. Proceedings of international test conference pp 1120–1129. doi: 10.1109/TEST.2002.1041869
76. Sedcole P, Cheung PYK (2006) Within-die delay variability in 90 nm FPGAs and beyond. In: Proceedings of the international conference on field-programmable technology, pp 97–104
77. Sedcole P, Cheung PYK (2008) Parametric yield modeling and simulations of FPGA circuits considering within-die delay variations. Trans Reconfig Technol Syst 1(2). doi: 10.1145/1371579.1371582
78. Sinha SK, Kamarchik PM, Goldstein SC (2000) Tunable fault tolerance for runtime reconfigurable architectures. In: Proceedings of the IEEE symposium on field-programmable custom computing machines, pp 185–192
79. Sivaswamy S, Bazargan K (2008) Statistical analysis and process variation-aware routing and skew assignment for FPGAs. Trans Reconfig Technol Syst 1(1):1–35. doi: http://doi.acm.org/10.1145/1331897.1331900
80. Snider G, Kuekes P, Williams RS (2004) CMOS-like logic in defective, nanoscale crossbars. Nanotechnology 15:881–891
81. Snider GS, Williams RS (2007) Nano/CMOS architetures using a field-programmable nanowire interconnect. Nanotechnology 18(3)
82. Steiner N (2008) Autonomous computing systems. Ph.D. thesis, Virginia Polytechnic Institute and State University. Available Online: 10.1109/AERO.2009.4839512
83. Steiner N, Athanas P (2009) Hardware autonomy and space systems. In: Proceedings of the IEEE aerospace conference, pp 1–13
84. Strukov DB, Likharev KK (2005) CMOL FPGA: a reconfigurable architecture for hybrid digital circuits with two-terminal nanodevices. Nanotechnology 16(6):888–900
85. Sverdlov VA, Walls TJ, Likharev KK (2003) Nanoscale silicon MOSFETs: a theoretical study. IEEE Trans Electron Devices 50(9):1926–1933
86. Tam S, Limaye RD, Desai UN (2004) Clock generation and distribution for the 130-nm Itanium 2 processor with 6-MB on-die L3 cache. IEEE J Solid State Circuits 39(4): 636–642
87. Trimberger SM (2008) Utilizing multiple test bitstreams to avoid localized defects in partially defective programmable integrated circuits. US Patent 7,424,655
88. Wang C, Hu Y, Lieber CM, Sun S (2008) Ultrathin Au nanowires and their transport properties. J Am Chem Soc 130: 8902–8903
89. Wells RW, Ling ZM, Patrie RD, Tong VL, Cho J, Toutounchi S (2004) Applicationspecific testing methods for programmable logic devices. US Patent 6,817,006
90. Whang D, Jin S, Lieber CM (2003) Nanolithography using hierarchically assembled nanowire masks. Nanoletters 3(7):951–954
91. Whang D, Jin S, Wu Y, Lieber CM (2003) Large-scale hierarchical organization of nanowire arrays for integrated nanosystems. Nanoletters 3(9):1255–1259
92. Williams S, Kuekes P (2001) Demultiplexer for a molecular wire crossbar network. US Patent 6,256,767

93. Wong JSJ, Sedcole P, Cheung PYK (2009) Self-measurement of combinatorial circuit delays in FPGAs. Trans Reconfig Technol Syst 2(2):1–22
94. Wu W, Jung GY, Olynick D, Straznicky J, Li Z, Li X, Ohlberg D, Chen Y, Wang S-Y, Liddle J, Tong W, Williams RS (2005) One-kilobit cross-bar molecular memory circuits at 30-nm half-pitch fabricated by nanoimprint lithography. Appl Phy A 80:1173–1178
95. Xilinx, Inc. (2005) 2100 Logic Drive, San Jose, CA 95124 Xilinx Virtex-II Pro and Virtex-II Pro X Platform FPGAs: Complete Data Sheet. DS083 http://direct.xilinx.com/bvdocs/publications/ds083.pdf
96. Xilinx, Inc., 2100 Logic Drive, San Jose, CA 95124 (2005) Virtex FPGA Series Configuration and Readback. XAPP 138 http://www.xilinx.com/bvdocs/appnotes/xapp138.pdf
97. Xilinx, Inc., 2100 Logic Drive, San Jose, CA 95124 (2008) Virtex-5 FPGA Configuration User Guide. UG191 http://www.xilinx.com/bvdocs/userguides/ug191.pdf
98. Yang C, Zhong Z, Lieber CM (2005) Encoding electronic properties by synthesis of axial modulation-doped silicon nanowires. Science 310:1304–1307
99. Yu G, Cao A, Lieber CM (2007) Large-area blown bubble films of aligned nanowires and carbon nanotubes. Nat Nanotechnol 2(6):372–377. doi: 10.1038/nnano.2007.150
100. Zhao W, Cao Y (2006) New generation of predictive technology model for sub-45 nm early design exploration. IEEE Trans Electron Devices 53(11):2816–2823

Summary

Power and variability have emerged as two major design issues in the nanometer technology regime. All components of complex modern ICs – logic and memory circuits, digital and analog – suffer form these issues. At system level, these issues are relevant for microprocessors, digital signal processing systems, reconfigurable hardware such as FPGA, as well as mixed-signal systems. In order to design low-power and robust electronic system using nanometer silicon technologies, it is important to understand the effects of parameter variations on low power designs and employ low-overhead design solutions to alleviate them. The design solutions can span different levels of design abstractions. The book has provided comprehensive coverage on design issues and solutions to satisfy the contradictory requirements of low power operation and robustness under parameter variations. Understanding of these issues and use of effective solutions would be key to continued scaling of CMOS. We hope that the all readers – students, researchers and practitioners – will be able to achieve a holistic view of the design challenges and low-cost solutions for low-power and variation-tolerant ICs.

The best results with respect to power dissipation and robustness can often be obtained by considering application-specific features. For example, DSP and multimedia systems are characterized by their elasticity to tolerate failures in certain components. One can use this knowledge to trade off power versus quality – or achieve graceful degradation in quality under temporal variations. Similarly, for multi-core systems with homogeneous cores, one can perform dynamic task migration or core-hopping to tolerate variation effects under power constraints. For analog cores or FPGA, similar application-specific optimizations can also be very effective. The book covers this important aspect in relevant chapters.

The modeling, analysis and design solutions presented in this book are expected to provide valuable resources for CAD tool developers, who can build appropriate design automation tools – technology specific or independent – for design analysis and optimization. With increasing complexity of modern integrated circuits, design techniques which are easily amenable for automation using CAD tool, will have definite advantages for practical applications.

With growing use of electronics in new areas, there will be new challenges as well as opportunities in terms of low-power and variation-tolerant design. One such

S. Bhunia, S. Mukhopadhyay (eds.), *Low-Power Variation-Tolerant Design in Nanometer Silicon*, DOI 10.1007/978-1-4419-7418-1,

example is biomedical circuits and system. Many biomedical circuits e.g. circuits used in implantable systems for monitoring biomedical signals (e.g. neuronal spikes, bladder pressure) would greatly benefit from ultralow power and long-term robust operation. Recent research shows that these applications can exploit the nature of the signal processing algorithms to drastically reduce power dissipation while maintaining robustness. The book can create pathways to employ the design analysis and optimization solutions in emerging applications.

With CMOS rapidly moving towards the end of its road-map due to fundamental physical limits, many alternative device technologies with interesting switching characteristics are emerging. These technologies show promises in terms of tera-scale integration density and tera-hertz performance. We anticipate, in the short term, these technologies will be used to hybridize with CMOS to address its limitations. In the long term, CMOS is likely to be replaced by a new technology. At this point we are unaware which of the emerging technologies will emerge as winner. We, however, believe the issues of power and power density as well as variation induced reliability and yield concerns will remain prevalent in the nanometer regime irrespective of the device technologies. Some of technologies, which can potentially replace CMOS, already show large device parameter variations due to process imperfections. Hence, the design solutions for low-power and variation tolerance presented in this book are expected to remain very relevant in future non-silicon technologies.

Index