MINIMIZING AND EXPLOITING LEAKAGE IN VLSI

A Dissertation

by

NIKHIL JAYAKUMAR

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2007

Major Subject: Computer Engineering

MINIMIZING AND EXPLOITING LEAKAGE IN VLSI

A Dissertation

by

NIKHIL JAYAKUMAR

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	Sunil P. Khatri
Committee Members,	Gwan Choi
	Prasad Enjeti
	Hong Liang
	Gianfranco Gerosa
	Fabio Somenzi
Head of Department,	Costas N. Georghiades

May 2007

Major Subject: Computer Engineering

ABSTRACT

Minimizing and Exploiting Leakage in VLSI. (May 2007)

Nikhil Jayakumar, B.S., University of Madras, India;

M.S., University of Colorado at Boulder

Chair of Advisory Committee: Dr. Sunil P. Khatri

Power consumption of VLSI (Very Large Scale Integrated) circuits has been growing at an alarmingly rapid rate. This increase in power consumption, coupled with the increasing demand for portable/hand-held electronics, has made power consumption a dominant concern in the design of VLSI circuits today. Traditionally dynamic (switching) power has dominated the total power consumption of VLSI circuits. However, due to process scaling trends, leakage power has now become a major component of the total power consumption in VLSI circuits. This dissertation explores techniques to reduce leakage, as well as techniques to exploit leakage currents through the use of sub-threshold circuits.

This dissertation consists of two studies. In the first study, techniques to reduce leakage are presented. These include a low leakage ASIC design methodology that uses high V_T sleep transistors selectively, a methodology that combines *input vector control* and circuit modification, and a scheme to find the optimum reverse body bias voltage to minimize leakage.

As the minimum feature size of VLSI fabrication processes continues to shrink with each successive process generation (along with the value of supply voltage and therefore the threshold voltage of the devices), leakage currents increase exponentially. Leakage currents are hence seen as a necessary evil in traditional VLSI design methodologies. We present an approach to turn this problem into an opportunity. In the second study in this dissertation, we attempt to exploit leakage currents to perform computation. We use sub-threshold digital circuits and come up with ways to get around some of the pitfalls associated with

sub-threshold circuit design. These include a technique that uses body biasing adaptively to compensate for Process, Voltage and Temperature (PVT) variations, a design approach that uses asynchronous micro-pipelined Network of Programmable Logic Arrays (NPLAs) to help improve the throughput of sub-threshold designs, and a method to find the optimum supply voltage that minimizes energy consumption in a circuit.

To my parents, my brother and my grandparents

ACKNOWLEDGMENTS

This dissertation and the work presented in it, has been possible thanks to the support of many people.

I would first like to thank my advisor Dr. Sunil P. Khatri. I couldn't have dreamed of having a better advisor. I know of no-one else with such sincerity and dedication to his profession and his students. He has always been full of fresh ideas, and it has been a joy to work with him. To me, he has been not just an advisor, but also a friend.

I thank all the members of my dissertation committee - Dr. Gwan Choi from the Computer Engineering group, Dr. Prasad Enjeti from the Power Electronics group, Dr. Helen Liang from Mechanical Engineering, Dr. Fabio Somenzi from the University of Colorado at Boulder and Dr. Gianfranco Gerosa from Intel, Austin. I would also like to thank Dr. Jean-Francois Chamberland from the Wireless Communications group for agreeing, at short notice, to substitute for a committee member at my final defense.

During my graduate and doctoral studies, I have had the pleasure of interacting and working with several very bright people. From my days at Boulder, I wish to thank Mitra Purandare, Nikhil Saluja, Arjun Kapoor, Ed Hursey, Chao Wang, Bing Li, Mohammed Awedh, HoonSang Jin, Sankar Gurumurthy. Vijay Nagarajan, Dr. Olgica Milenkovic and Dr. Fabio Somenzi. A special thanks to Dr. Fabio Somenzi with whom I published my first paper. Dr. Somenzi's patience and his humility is something I admire greatly.

From my days here at Texas A&M, I have been fortunate to work with Kanupriya Gulati, Rajesh Garg, Jeff Cobb, Ganesh Venkatraman, Mandar Waghmode, Vijay Balasubramanian, Seraj Ahmed, Karandeep Singh, Suganth Paul, Charu Nagpal and Dr. Gwan Choi. I'd like to specially thank Kanupriya Gulati and Rajesh Garg who have helped in my research and in the work presented in this dissertation.

During my internships at National Semiconductor in Longmont, Colorado and at Intel

in Austin, Texas, I learned a lot from several people. At National Semiconductor, my manager, Dr. Sandeep Dhar gave me a lot of independence and let me work on several interesting topics including reverse body biasing to reduce leakage which is a part of this dissertation. My internship at Intel was an enjoyable and fruitful experience thanks to all the help and support from Hans Yeager, Steve Curtis, Gus Yeung, Hasan Taufique, Manju Shamanna and Gianfranco Gerosa. I'd like to specially thank Dr. Gianfranco Gerosa who agreed to be on my dissertation committee. His insights on how things function in industry and how to manage a group were invaluable to me. A special thanks also to Hans Yeager with whom I had several discussions on power reduction techniques and to Gus Yeung who helped me get a taste of reliability issues in VLSI.

In the course of my doctoral studies, I have had to put in quite a few long hours. Fortunately, I had a great set of friends to help me unwind after a long hard day. I want to specially thank my room-mates here at College Station - Arun Venugopal and Vikram Ramakanth. Their remarkable culinary skills are legendary among the Indian students at Texas A&M.

I am also very thankful for all the support and encouragement I have received from my family. I am thankful to have had the company of my younger brother Rohan while growing up in India. I was fortunate to have my grandparents (my mother's parents) close to me while I grew up, and they influenced me a great deal in my life. All grandparents, they say, pamper their grandchildren. My grandmother has four grandchildren so this pampering had to be shared, but she made each one of us feel special and loved. My grandfather in particular influenced my thinking. A believer in rational thinking, he taught me to never blindly believe everything I was told, but to question and find out the answers for myself. I still have discussions with him on a wide range of topics - from the existence of god, to the latest advances in genetics - from quantum physics to the illusion of freewill. I admire him for his wisdom and am thankful to have had his influence in my life. Lastly, but by

no means the least, I thank my parents. My mother influenced my character a great deal. Right from a very young age my mother taught me to do my best at anything I did. I had the freedom to do what I was most interested in as long as I did my best. But above all, she taught my brother and me the importance of being a good human being. She told us to be ambitious and chase our dreams, but make sure that we had our feet firmly planted to the ground. My father ensured that I had everything I needed to chase my dreams. Being a Chartered Accountant by profession, my father was careful with the way he spent money. It was almost impossible to convince him to buy me the latest game system, but if I asked him for a book, I'd get it straightaway. He spared no expense in making the best possible education available to my brother and me. I am extremely fortunate to have had parents like my mother and father.

The work presented in this dissertation would not have been possible without the tremendous amount of help and encouragement I have received from my family, my friends, my colleagues, my teachers and my advisor. With their continued support, I am confident I can achieve much more.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

TABLE Page

LIST OF FIGURES

FIGURE Page

FIGURE Page

CHAPTER I

INTRODUCTION

Semiconductor manufacturers have managed to faithfully follow Moore's law and hence, have been able to integrate more transistors and circuit functionality on a single chip with each process generation. However, the relentless march towards smaller, faster and cheaper VLSI (Very Large Scale Integrated Circuit) chips, has contributed to the excessive power consumption of many circuits used today. In this chapter, we discuss the reasons for this rapid growth in power consumption.

I-A. The Need for Low Power Design

Since the advent of CMOS technology, an increased number of transistors per die and greater performance have been the primary driving factors for the semiconductor industry and process technology. The ability to integrate more transistors per die allowed chip manufacturers to put more components of a system into a single package and thus reduce not just the sizes of the electronic devices we use today but also the cost and delay. The intense competition in the semiconductor industry has made chip manufacturers chase these goals aggressively. To the credit of the semiconductor industry, these goals (more transistors per die and greater performance) have been growing at an exponential rate, following Moore's law. However, in the process, the power dissipation of the Integrated Circuit (IC) has been growing at an alarming rate as well. In recent times, the excessive power consumption of contemporary circuits has become a dominant design concern [4]. In addition to shortened battery life for portable electronics, higher power consumption results in aggravated on-chip temperatures, which can result in a reduced operating life for the IC [5].

This dissertation follows the style of the *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.

For portable electronics, longer battery life is the most important design constraint. As a result, low power consumption becomes a crucial requirement for circuits used in portable electronics. In fact, the rapid growth in the demand for portable electronics is one of the major drivers that has forced semiconductor manufacturers to make conscious efforts to reduce power consumption.

However, power consumption is not an issue just for portable electronics today. ICs that consume more power also dissipate more heat and this necessitates more expensive cooling solutions. In fact, the use of liquid cooling in high performance desktop computers is now fairly common (especially in the gamer's market). In the consumer market, saving even a few cents per part can translate into significant profits for a company. Hence, an IC that dissipates a lot of heat and thus requires an expensive cooling solution directly impacts the cost of a system using the IC. For organizations that employ large server farms, the cost of cooling the servers and the power consumption of the servers themselves are significant, especially in this day and age of rising energy costs.

Hence, low power consumption is a zero order constraint for most ICs manufactured today. In fact, *higher performance-per-watt* is the new mantra for micro-processor chip manufacturers today.

I-B. Leakage and Its Contribution to IC Power Consumption

The power consumption of a VLSI chip is broadly classified into two - dynamic power and leakage power. Dynamic power is also often referred to as active power or switching power. This is the power consumed when a transistor switches, transferring charge. Since this charge transfer is required for any computation, this source of power dissipation is often considered a more useful or necessary source of power dissipation.

On the other hand, leakage power is considered a wasteful expenditure of power. Leak-

age power is the power consumed when a *turned off* device leaks current. This source of power consumption is considered wasteful expense and is the dominant source of power dissipation in many portable electronic devices (such as cell-phones, PDAs etc.) that spend most of their time in the *standby* state.

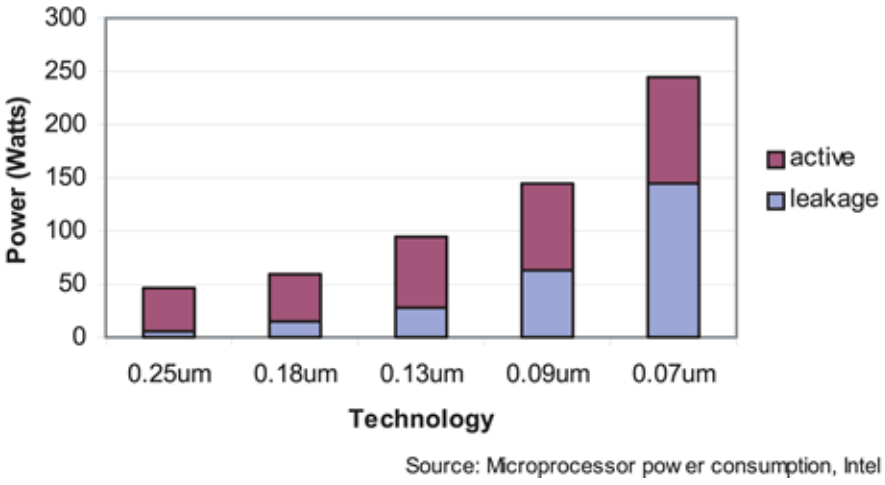


Fig. I.1. Recent power trends [1]

As can be seen from Figure I.1 [1], IC power consumption has been increasing rapidly as we move to new technology nodes. Interestingly, while both dynamic as well as leakage power have been increasing, the leakage power component has been growing at a significantly faster rate. The reason for this trend is explained below.

Consider the n-channel MOS (NMOS) device. An NMOS device has four terminals, the drain, gate, source and bulk, and it operates in one of three modes of conduction [6, 7], depending on the voltage of its terminals (V_d,V_g, V_s, V_b respectively). In the equations that follow,

$V_{xy} = V_x - V_y$.

- Sub-threshold region :

$I_{ds}^{sub} = \frac{W}{L} I_{D0} e^{(\frac{V_{gs}-V_T-V_{off}}{nv_t})} [1 - e^{-\frac{V_{ds}}{v_t}}]$

when $V_{gs} < V_T$

- Linear (Triode) region :

  $I_{ds}^{lin} = \beta([V_{gs} - V_T]\, V_{ds} - \frac{V_{ds}^2}{2})$

  when $0 < V_{ds} < V_{gs} - V_T$

- Saturation region :

  $I_{ds}^{sat} = \frac{\beta}{2}(V_{gs} - V_T)^2$

  when $0 < V_{gs} - V_T < V_{ds}$

The equations above express the current I_{ds} through an NMOS transistor in the three modes of conduction. In the above equations, V_T is the device *threshold voltage*. It depends on process dependent factors like gate and insulator materials, thickness of insulator and channel doping density. It also depends on operational factors like V_{sb} (body effect)[1] and temperature (V_T is inversely proportional to device junction temperature). V_T is typically engineered to be about 20-25% of VDD. Also, $\beta = (\mu\varepsilon/t_{ox}) \cdot (W/L)$ where μ is the surface mobility of electrons (holes for a PMOS device) in the channel, ε is the permittivity of the gate oxide[2] and t_{ox} is the gate oxide thickness. W and L are the device width and length. Also, I_{D0} is a constant while $v_t = kT/q$. Here k is the Boltzman constant, q is the charge of an electron and $v_t = 26mV$ at room temperature. n is the sub-threshold swing parameter (a constant). Finally, V_{off} is a constant, typically equal to -0.08V.

With technology scaling, supply voltages have been scaling down as well. The switch-

[1] Body effect increases the threshold voltage of a device based on the following equation:
$V_T = V_T^0 + \gamma(\sqrt{|(-2)\phi_F + V_{sb}|} - \sqrt{|2\phi_F|})$, where V_T^0 is the threshold voltage at zero V_{sb}, γ is the body-effect coefficient - a physical parameter that expresses the impact of changes in V_{sb} and ϕ_F is the Fermi potential (typically 0.3V for silicon)

[2] $\varepsilon = k \cdot \varepsilon_0$, where $k \equiv$ dielectric constant of the gate oxide.

ing delay of a device is dictated by the current that can flow through it when the device is *turned on* (the device is in the saturation region). From the equation for the current of a device in the saturation region, it is clear that, to maintain a high saturation current and hence a small delay, any decrease in the supply voltage (which implies a decrease in V_{gs}), has to be accompanied by a decrease in the threshold voltage V_T of the device as well.

The leakage current for a PMOS or NMOS device corresponds to the I_{ds} of the device when the device is in the *cut-off* or *sub-threshold* region of operation. From the equation for I_{ds} in the sub-threshold region, we can see that the leakage current is exponentially dependent on the threshold voltage of the device. This is why a reduction in supply voltage (which is accompanied by a reduction in threshold voltage) results in exponential increase in leakage. Hence, with technology scaling and its accompanying supply voltage reduction, the leakage power consumption has been growing at a much faster rate than dynamic power consumption, as indicated in Figure I.1.

Another contributor to the greater rate of increase in leakage power is the fact that more logic is being integrated onto a single die. During operation however, there are only a few portions of the chip performing useful computations while a majority of the chip simply leaks, wasting power.

The power consumed by a design in the standby mode of operation is due to leakage currents in its devices. While the sub-threshold leakage current I_{ds}^{sub} is the major component of leakage (in typical CMOS usage scenarios) there are several other sources of leakage as well. Figure I.2 (adapted from [2]) shows the various sources of leakage for an NMOS device. In Figure I.2, I_{tox} represents the oxide tunneling current through the gate of the device, while I_{hot-e} represents the gate leakage due to *hot-carriers* (electrons with high energy due to the applied electric field) being injected into the oxide layer of the gate. Gate leakage current is mainly due to these two components. The currents I_{pn} and I_{BTBT} are the currents that flow through the reverse-biased *pn* junction formed at the edges of the

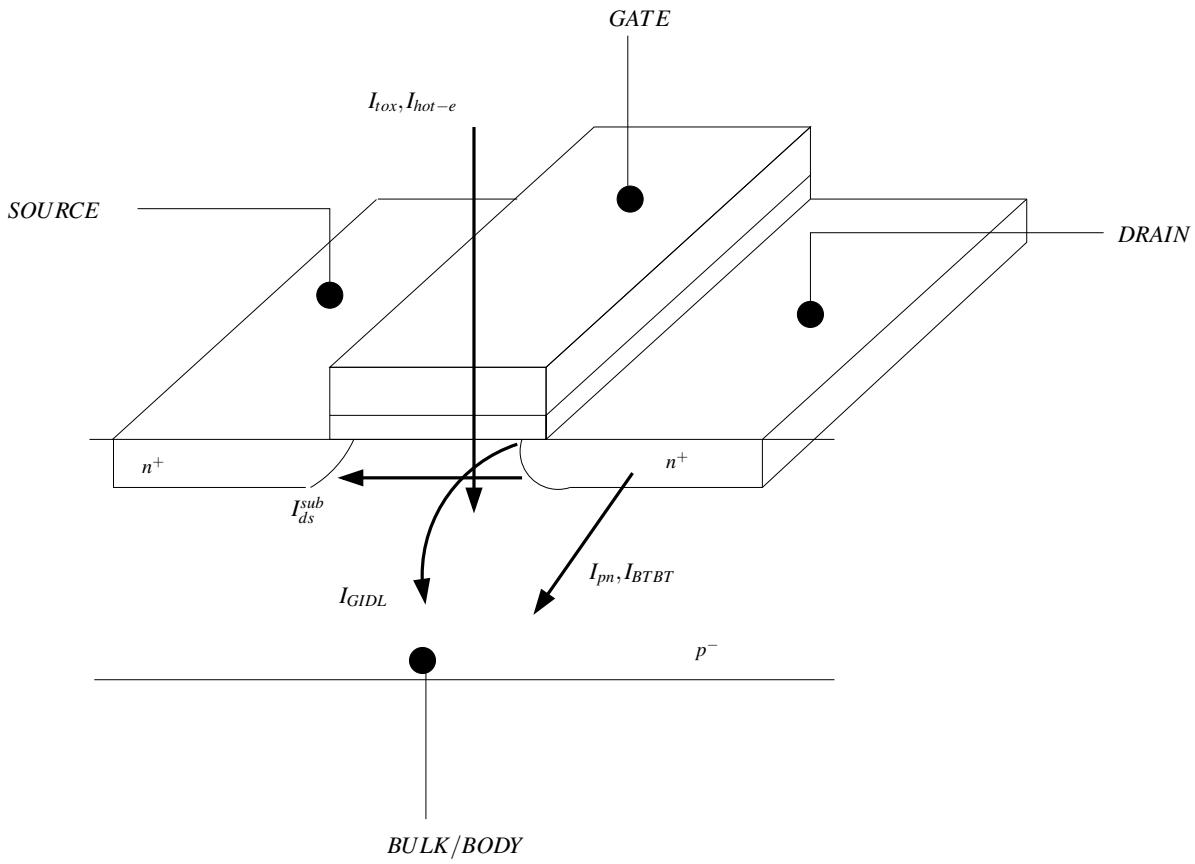


Fig. I.2. Sources of leakage (NMOS device) (adapted from [2])

bulk and drain of the device. I_{pn} consists of mainly two components - a minority carrier diffusion/drift current and a current due to electron-hole pair generation. I_{BTBT} is the band-to-band tunneling (BTBT) current which is a current due to the tunneling of electrons from the valence band of the p-region (from the bulk) to the conduction band of the n-region (to the drain). This tunneling happens due to a high electric field across the bulk-drain junction (which can happen when a Reverse Body Bias (RBB) is applied). BTBT current is also referred to as bulk-BTBT or Gate Edge Drain Leakage (GEDL). I_{GIDL} is the Gate Induced Drain Leakage current (GIDL) which is also referred to as surface BTBT. This current occurs when the gate bias is negative relative to the drain. Under most operating

scenarios and for most CMOS devices used today it is the sub-threshold leakage from the drain to the source of a device that dominates total leakage. In some situations (such as when there is a reverse body bias applied), the BTBT component may dominate. Due to process scaling trends (shrinking of gate oxide thickness) gate leakage has also become a concern. However, there is very little (apart from keeping supply and gate voltages low) that can be done at the design stage to tackle gate leakage. It is expected that the gate leakage issue would be tackled at the process technology stage.

With the prevalence of portable electronics, it is crucial to keep the leakage currents of a design small in order to ensure a long battery life in the standby mode of operation.

I-C. Dissertation Overview

So far we have seen how power consumption is a major concern in today's VLSI designs. More particularly, we have seen how leakage power is a significant component of the total power consumption of a chip.

This dissertation consists of two studies. The first study in this dissertation addresses leakage *reduction* approaches while the second explores techniques to *exploit* leakage currents to perform computation.

Since leakage power consumption is seen as a major issue in VLSI design today, there has been significant research into techniques to reduce leakage. In the first study, new techniques to reduce leakage are proposed. These include a design approach that uses high V_T *sleep transistors* selectively, a technique that modifies a circuit to reduce leakage while simultaneously finding the best input vector that minimizes leakage and a scheme to find the optimum reverse body biasing voltage to minimize leakage.

In the second study presented in this dissertation, we attempt to exploit leakage currents rather than minimize them. We propose the use of sub-threshold digital circuits and

come up with ways to get around some of the pitfalls associated with sub-threshold circuit design. These include a self-adjusting adaptive body-biasing technique that helps make a sub-threshold circuit less sensitive to Process, Voltage and Temperature (PVT) variations, a design approach that helps improve the throughput of sub-threshold designs through the use of asynchronous micro-pipelined Network of Programmable Logic Arrays (NPLAs), and a method to find the optimum supply voltage that minimizes energy consumption in a circuit.

I-D. Dissertation Outline

This dissertation is organized as follows. In Chapter II, we discuss some previous leakage reduction approaches. In particular, we discuss Power-gating/MTCMOS techniques, Body biasing and Input Vector Control. The advantages and disadvantages of each of these techniques are also discussed in this chapter.

In Chapter III, we describe a new low-leakage standard cell based (Application Specific Integrated Circuit) ASIC design methodology that we call the "HL" methodology. This "HL" methodology is based on ensuring that during standby operation, the supply voltage is applied across more than one *off* device and there is at least one *off* device in the leakage path which has a high V_T. For each standard cell in a library, we design two low leakage variants, If the inputs of a cell during the standby mode of operation are such that the output has a high value, we use the variant that minimizes leakage in the pull-down network. Similarly we use the variant that minimizes leakage in the pull-up network if the output has a low value. While technology mapping a circuit, we determine the particular variant to utilize in each instance, so as to minimize the leakage of the final mapped design. We present experimental results that compare placed-and-routed area, leakage and delays of this new methodology against MTCMOS and a regular standard cell based design

style. The results show that our new methodology has better speed and area characteristics than MTCMOS implementations. The leakage current for HL designs can be dramatically lower than the worst-case leakage of MTCMOS based designs, and two orders of magnitude lower than the leakage of traditional standard cells. In contrast to the leakage of an MTCMOS design, the HL approach yields precisely estimable leakage values.

In Chapter IV, we present an approach which minimizes leakage by simultaneously modifying the circuit while deriving the input vector that minimizes leakage. This approach involves traversing a given circuit topologically from inputs to outputs, and replacing gates to set as many gates as possible to their low leakage state (in the sleep/standby state). The replacement does not necessarily reduce the leakage of the gate g being replaced, but helps set the gates in the transitive fanout of g to their low leakage states. Gate replacement is performed in a slack-aware manner, to minimize the resulting delay penalty. One of the major advantages of this technique is that we achieve a significant reduction in leakage without increasing the delay of the circuit.

In Chapter V, we first present results (from a 130nm test-chip) that prove that while sub-threshold leakage current decreases with applied Reverse Body Bias (RBB), another leakage component, the bulk Band-to-Band-Tunneling (BTBT) leakage component actually increases with applied RBB. We find that, there exists an optimum RBB which minimizes *total* leakage. We present a scheme that monitors the total leakage of a transistor and identifies the optimum RBB voltage that minimizes total leakage. Our method consists of a leakage current monitor, and a digital block that senses the discharging (charging in the case of a PMOS transistor) of a representative *leaking* NMOS device in the design. Based on the speed of discharge, which is faster for leakier devices, an appropriate RBB value is applied. The scheme presented incurs very reasonable placed-and-routed area and power penalties in its operation.

In Chapter VI, we introduce the idea of operating circuits in the sub-threshold region

of operation. We present exploratory studies that reveal the opportunity that sub-threshold circuits offer. We also list some of the disadvantages of sub-threshold circuit design, along with scenarios where such a methodology could be applied.

Chapter VII presents a sub-threshold design methodology which dynamically compensates for inter and intra-die process, supply voltage and temperature (PVT) variations. This compensation is achieved by performing bulk voltage adjustments in a closed-loop fashion. Our design methodology uses a multi-level network of medium sized Programmable Logic Arrays (PLAs) as the circuit implementation structure. The design has a global *beat clock* to which the delay of a spatially localized cluster of PLAs is ”phase locked”. The synchronization is performed in a closed-loop fashion, using a phase detector and a charge pump which drives the bulk nodes of the PLAs in the cluster. We demonstrate the ability of our technique to dynamically phase lock the PLA delays to the beat clock, across a wide range of PVT variations, enabling significant yield improvements. Without the approach of this chapter, the high sensitivity of the sub-threshold current to PVT variations would make sub-threshold circuit design untenable.

In Chapter VIII, we first prove that while a lower voltage does result in lower *power* consumption, it does not translate to a lower *energy* consumption. In fact, we find that the optimum voltage to minimize energy consumption depends on the circuit topology. We describe a technique to find the energy optimum VDD value for a design, and show that for minimum energy consumption, the circuit may need to be operated at VDD values which are above the NMOS threshold voltage value. We study this problem in the context of designing a circuit using a network of dynamic NOR-NOR PLAs.

In Chapter IX, we propose an approach to try to reduce the speed gap between sub-threshold and traditional designs. We propose a sub-threshold circuit design approach based on asynchronous micropipelining of a levelized network of PLAs. We demonstrate that by using our approach, a design can be sped up by about $7\times$, with an area penalty

of 47%. Further, our approach yields an energy improvement of about 4×, compared to a traditional Network of PLA based design.

Finally, in Chapter X we concludes this dissertation. We present some future directions for research and a summary of the broader impact of this work.

I-E. Chapter Summary

In this chapter, we have introduced the power consumption problem faced in VLSI design today. In particular we have discussed why leakage power consumption is a major concern for today's designs. Starting with the next chapter, we discuss techniques to minimize leakage, followed by approaches to exploit leakage through the use of sub-threshold circuits.

CHAPTER II

EXISTING LEAKAGE MINIMIZATION APPROACHES

II-A. Chapter Overview

As mentioned in Chapter I, leakage is a major concern for VLSI design today. Hence, there has been significant research into techniques to reduce leakage. In this chapter, we discuss some previous approaches to reduce leakage for digital VLSI designs.

II-B. Leakage Minimization Approaches - An Overview

In recent times, leakage power reduction has received much attention in academia as well as industry. Several means of reducing leakage power have been proposed. Some of these are mentioned here.

II-B.1. Power Gating / MTCMOS

One of the natural techniques to reduce the leakage of a circuit is to gate the power supply using power-gating transistors (also called *sleep* transistors). Typically high-V_T power-gating transistors are placed between the power supplies and the logic gates. This is called the MTCMOS (Multi-Threshold CMOS) approach [8, 9]. In standby, these power-gating transistors are turned off, thus shutting off power to the gates of the circuit. The MTCMOS approach can reduce circuit leakages by up to 2 to 3 orders of magnitude (depending on the threshold voltages and size of the sleep transistors used). However, the addition of sleep transistors causes an increase in the delay of the circuit. This delay penalty can be reduced by appropriately sizing up the sleep transistor. The downside to the up-sizing of the sleep transistor is the accompanied increase in the time and switching energy spent in *waking up* the circuit. As a consequence, power-gating (turning off the sleep transistors) is applied

only when the circuit is expected to be in the standby state for a long period of time and when the wake-up time is tolerable. If a circuit using power-gating/sleep transistors goes in and comes out of the standby state too often, the power consumption may actually *increase* due to the higher power consumed in waking up the circuit. Another disadvantage of the MTCMOS approach is the fact that implementation of this technique requires circuit modification and possibly additional process steps (since high-V_T sleep transistors are used). Also, since cell inputs and outputs as well as bulk nodes float in an MTCMOS design operating in standby mode, the precise prediction or control of leakage is extremely difficult in MTCMOS. The voltage of these floating nodes can significantly affect the device threshold voltages. Hence, it is very difficult to precisely predict or control leakage in MTCMOS designs. Another drawback of MTCMOS is that memory elements in MTCMOS would require clean power supplies routed to them if we want to maintain their state in standby mode [9].

There has also been some research into the sizing of these sleep transistors. A conservative method to sizing the sleep transistors would be to first estimate the width of the sleep transistor required for each gate (or standard-cell) in a design such that the delay of the individual gate is within a specified bound and then add up the sleep transistor widths for all gates to come up with the total sleep transistor width required. In [8] the authors propose a MTCMOS standby device sizing algorithm which is based on mutually exclusive discharging of gates. This technique is hard to utilize for random logic circuits as opposed to the extremely regular circuits which are used as illustrative examples in [8].

In [10], an MTCMOS-like leakage reduction approach was proposed, in which the MTCMOS sleep devices are connected in parallel with diodes. This ensures that the supply voltage across the logic is $VDD - 2V_D$, where V_D is the forward-biased voltage drop of a diode. The sub-threshold leakage current is significantly larger when $V_{ds} \gg nv_t$. This is because V_T drops due to the DIBL (Drain Induced Barrier Lowering) effect when V_{ds} is

large [7]. The approach of [10] ensures that the V_{ds} across the sleep transistors is limited to $VDD - 2V_D$, thus keeping the sub-threshold leakage current low.

II-B.2. Body Biasing / VTCMOS

Increasing V_T via body effect and bulk voltage modulation is another way to reduce leakage power. The leakage current of a transistor decreases with greater applied Reverse Body Bias. Reverse Body Biasing affects V_T through body effect, and sub-threshold leakage has an exponential dependence on V_T as seen in the sub-threshold current equation (Equation 2.1).

$$I_{ds}^{sub} = \frac{W}{L} I_{D0} e^{\frac{V_{gs}-V_T-V_{off}}{nv_t}} [1 - e^{-\frac{V_{ds}}{v_t}}] \tag{2.1}$$

The body effect equation can be written as:
$V_T = V_T^0 + \gamma(\sqrt{|(-2)\phi_F + V_{sb}|} - \sqrt{|2\phi_F|})$, where V_T^0 is the threshold voltage at zero V_{sb}, γ is the body-effect coefficient - a physical parameter that expresses the impact of changes in V_{sb} and ϕ_F is the Fermi potential (typically 0.3V for silicon). Thus the threshold voltage of devices can be dynamically adjusted using body biasing. Hence, this method of controlling the threshold voltage of transistors through body biasing is often referred to as the Variable Threshold CMOS or VTCMOS technology.

In [11], the authors describe how they applied VTCMOS technology to both the logic and memory elements of a 2-D Discrete Cosine Transform (DCT) core processor. During the active mode of operation, they apply a reverse body bias of 0.5V and during standby they increase the reverse body bias to 3.3V. The VTCMOS scheme implemented consisted of leakage current monitors (LCMs) to monitor the sub-threshold leakage and two charge-pump circuits - one to increase the applied RBB and another to decrease the applied RBB. These charge-pumps were controlled in a closed-loop fashion using the leakage current monitors for feedback. In [12], the authors study the characteristics of VTCMOS for series

connected circuits. They find that VTCMOS is effective for improving the performance of series connected devices too. In [13], the authors propose a compact analytical model of VTCMOS to help study the currents through a VTCMOS transistor during the active and standby states. They also study the influence of short channel effect (SCE) on the performance of VTCMOS.

The advantage with VTCMOS is that leakage current can be reduced in the standby mode by applying a reverse body bias (RBB) which raises the threshold voltage or the delay can be reduced in the active mode by applying a forward body bias which decreases the threshold voltage. However, with current technology scaling, the body-effect coefficient γ is reducing. Apart from this, there is also the overhead of implementing additional body-biasing supplies and the need to use special processes (such as the triple-well process) in order to provide separate well biasing. This method offers the advantage of decreasing the leakage in standby mode while not increasing the delay in the active mode.

In [14], the authors propose a dynamic threshold MOSFET design for low leakage applications. In this scheme, the device gate is connected to the bulk, resulting in high-speed switching and low leakage currents through body effect control. The drawback of this approach is that it is only applicable in situations where VDD is lower than the diode turn-on voltage. Also, the increased capacitance of the gate signal slows the device down, and as a result, the authors propose the use of this technique for partially depleted SOI (Silicon-On-Insulator) designs.

II-B.3. Input Vector Control

Another technique used to minimize leakage is the technique of *parking* a circuit in its minimum leakage state. This technique takes advantage of the fact that the leakage of a gate is dependent on the state of the inputs of the gate. The technique involves very little or no circuit modification and does not require additional power supplies. A combinational

circuit is *parked* in a particular state by driving the primary inputs of the circuit to a particular value. In the standby mode , this value can be scanned in or forced using MUXes (with the standby/sleep signal used as a select signal for the MUX). This technique is frequently referred to as *input vector control*. Finding the best (lowest leakage) input vector, also called the Minimum Leakage Vector (MLV) determination problem, is known to be an NP-hard problem. However, several heuristics have been developed to find an optimal vector. In [15], the authors find a minimal leakage vector using random search with the number of vectors used for the random search selected to achieve a specified statistical confidence and tolerance. In [16], the authors reported a genetic algorithm based approach to solve the problem. The authors of [17] introduce a concept called leakage observability, and based on this idea, describe a greedy approach as well as an exact branch and bound search to find the maximum and minimum leakage bounds. The work of [18] is based on an ILP formulation. It makes use of pseudo-Boolean functions which are incorporated into an optimal ILP model and a heuristic mixed integer linear programming method as well. In [19, 20], the authors present an MDD [21] based algorithm to determine the lowest leakage state of a circuit. The use of MDD based MLV computations limits the applicability of [19] to small designs. In [20], an Algebraic Decision Diagram (ADD) based approach is presented to determine and implicitly represent the leakage value for all input vectors of a combinational circuit. In its exact form, the technique can compute the leakage value of each input vector. To broaden the applicability of the technique, an approximate version of the algorithm was presented as well. The approximation is done by limiting the total number of discriminant nodes in any ADD. The methods in [20] compute a leakage histogram for the design.

In [22], the authors present a greedy search based heuristic, guided by node controllabilities and functional dependencies. The algorithm used in [22] involves finding the controllability and the controllability lists of all nodes in circuit and then using this infor-

mation as a guide to choose gates to set to a low leakage state. The controllability of a node is defined as the minimum number of inputs that have to be assigned to specific states in order to force the node to a particular state (based on concepts used in automatic test pattern generation) [23]. Controllability lists are defined as the minimum constraints necessary on the input vector to force a node to particular state. The time complexity of their algorithm is reported to $O(n^2)$ where n is the number of cells (gates) in the circuit. However in estimating the complexity of their algorithm, it is not clear if the authors include the time taken to generate the controllabilities and controllability lists of each node in the circuit. While finding the controllabilities can be done fairly easily [23], generating the controllability lists can be more involved.

In [24], the authors express the problem of finding a minimum leakage vector as a satisfiability problem and use an incremental SAT solver to find the minimum and maximum leakage current. While their approach worked well for small circuits, the authors report very large runtimes for large circuits. The authors therefore suggest using their algorithm as a checker for the random search suggested in [15].

The approach of [25], utilizes approximate signal probabilities of internal nodes to aid in finding the minimum leakage vector. A probabilistic heuristic was used to select the next gate to be processed, as well as to select the best state of the selected gate. A fast SAT solver was employed to ensure the consistency of the assignments that were made in this process.

II-C. Chapter Summary

In this chapter, we have presented some existing approaches to leakage power reduction. In the next few chapters, we propose some new approaches to tackle the leakage reduction problem.

CHAPTER III

THE HL APPROACH - A LOW-LEAKAGE ASIC DESIGN METHODOLOGY

III-A. Chapter Overview

In this chapter we describe a new low-leakage standard-cell based ASIC design methodology. This methodology is based on the use of modified standard-cells, designed to reduce leakage currents (by almost two orders of magnitude) in standby mode, and also to allow a precise estimation of leakage current. For each cell in a standard-cell library, two low-leakage variants of the cell are designed. If the inputs of a cell during the standby mode of operation are such that the output has a high value, we minimize the leakage in the pull-down network Similarly we minimize leakage in the pull-up network if the output has a low value. In this manner, two low-leakage variants of each standard-cell are obtained. While technology mapping a circuit, we determine the particular variant to utilize in each instance, so as to minimize leakage of the final mapped design.

We present experimental results that compare placed-and-routed area, leakage and delay of this new methodology against MTCMOS and a regular standard-cell based design style. The results show that our new methodology (which we call the "HL" methodology) has better speed and area characteristics than MTCMOS implementations. The leakage current for HL designs can be dramatically lower than the worst-case leakage of MTCMOS based designs, and two orders of magnitude lower than the leakage of traditional standard-cells. An ASIC design implemented in MTCMOS would require the use of separate power and ground supplies for latches and combinational logic, while our methodology does away with such a requirement. Another advantage of our methodology is that the leakage is precisely estimable, in contrast with MTCMOS. The primary contribution in this chapter, is a new low leakage design style for static CMOS designs. In addition, we also discuss

techniques to reduce leakage in dynamic (domino logic) designs.

III-B. Philosophy of the HL Approach

The leakage current for a PMOS or NMOS device corresponds to the I_{ds} of the device when the device is in the *cut-off* or *sub-threshold* region of operation. The expression for this current [26] is:

$$I_{ds}^{sub} = \frac{W}{L} I_{D0} e^{\frac{V_{gs}-V_T-V_{off}}{nv_t}} [1 - e^{-\frac{V_{ds}}{v_t}}] \tag{3.1}$$

Here I_{D0} and V_{off} (typically $V_{off} = -0.08V$) are constants, while v_t is the thermal voltage (26mV at 300°K) and n is the sub-threshold swing parameter.

We note that I_{ds} increases exponentially with a decrease in V_T. This is why a reduction in supply voltage (which is accompanied by a reduction in threshold voltage) results in exponential increase in leakage.

Another observation that can be made from Equation 3.1 is that I_{ds} is significantly larger when $V_{ds} \gg nv_t$. For typical devices, this is satisfied when $V_{ds} \simeq VDD$. The reason for this is not only that the last term of Equation 3.1 is close to unity, but also that with a large value of V_{ds}, V_T would be lowered due to drain induced barrier lowering - DIBL (V_T decreases approximately linearly with increasing V_{ds}) [7, 26]. Therefore, leakage reduction techniques should ensure that the supply voltage is not applied across a single device, as far as possible.

Our approach to leakage reduction attempts to ensure that the supply voltage is applied across more than one *turned-off* device and one of those devices is a high-V_T device. This is achieved by selectively introducing a high-V_T PMOS or NMOS supply gating device in either the pullup network of a gate (if the output is low in standby) or the pulldown network of a gate (if the output is high in standby). By this design choice, we obtain

standard-cells with both *low and predictable* standby leakage currents, unlike MTCMOS based approaches.

III-C. Related Previous Work

Previous design approaches have suggested the use of dual-threshold devices [8] in an MTCMOS configuration MTCMOS utilizes NMOS and PMOS power supply gating devices. The authors propose a MTCMOS standby device sizing algorithm which is based on mutually exclusive discharging of gates. This technique is hard to utilize for random logic circuits as opposed to the extremely regular circuits which are used as illustrative examples in [8]. In [9], the authors describe an MTCMOS implementation of a PLL using a 0.5μm process. In both these works, the problem of estimating the leakage of an MTCMOS design is not addressed. In practice, the leakage of such a design can vary widely and is hard to control or predict. The threshold voltage is modified by bulk bias (via body effect) and DIBL, which are determined in part by the voltages of the bulk/source and source/drain nodes. Since cell inputs and outputs as well as bulk nodes float in an MTCMOS design operating in standby mode, precise prediction or control of leakage is impossible in MTCMOS. Cell input and output voltages affect the leakage of a gate as seen in Equation 3.1. The bulk voltage V_b affects V_T through body effect, and sub-threshold leakage has an exponential dependence of V_T as seen in Equation 3.1. Hence MTCMOS designs can have a large range of leakage currents, with little ability to predict or control the actual leakage current.

The threshold voltage of a device drops due to the DIBL (Drain Induced Barrier Lowering) effect when V_{ds} is large [7]. Hence, leakage can be limited by making sure that the V_{ds} across a turned-off device is limited. In [10], the authors present a technique that ensures that the entire supply voltage (VDD) is not applied across one device. They propose

an MTCMOS-like leakage reduction approach, in which the MTCMOS sleep devices are connected in parallel with diodes. This helps ensure that the V_{ds} across the sleep devices is no greater than $VDD - 2V_D$, where V_D is the forward-biased diode voltage drop.

III-D. The HL Approach

Our goal is to design standard-cells with *predictably* low leakage currents. To achieve this, we design two variants of each standard-cell. The two variants of each standard-cell are designated "H" and "L". If the inputs of a cell during the standby mode of operation are such that the output has a high value, we minimize the leakage in the pull-down network. So a *footer* device (a high-V_T NMOS with its gate connected to $\overline{standby}$) is used. We call such a cell the "H" variant of the standard-cell. Similarly, if the inputs of a cell during the standby mode of operation are such that the output has a low value, we minimize the leakage in the pull-up network by adding a *header* device (a high-V_T PMOS with its gate connected *standby*), and call such a cell the "L" variant of the standard-cell.

This exercise, when carried out for a NAND3 gate, yields the circuits shown in Figure III.1. Note that the MTCMOS circuit is also shown in this figure. Although the PMOS and NMOS supply gating devices (equivalently called *header* and *footer* devices (devices shown shaded in Figure III.1) are shown in the circuit for the MTCMOS design, such devices are in practice shared by all the standard-cells of a larger circuit block.

In our design approach, we utilized the same base standard-cell library for all design styles. Our standard-cell library consisted of INVA, INVB, NAND2A, NAND2B, NAND3, NAND4, NOR2, NOR3, NOR4, AND2, AND3, AND4, OR2, OR3, OR4, AOI21, AOI22, OAI21 and OAI22 cells. We utilized the *bsim100* predictive 0.1μm model cards [27]. The devices have a $V_T^N = 0.26V$ and $V_T^P = -0.30V$. The header and footer devices we utilized had $V_T^N = 0.46V$ and $V_T^P = -0.50V$. We sized the header and footer devices so that the

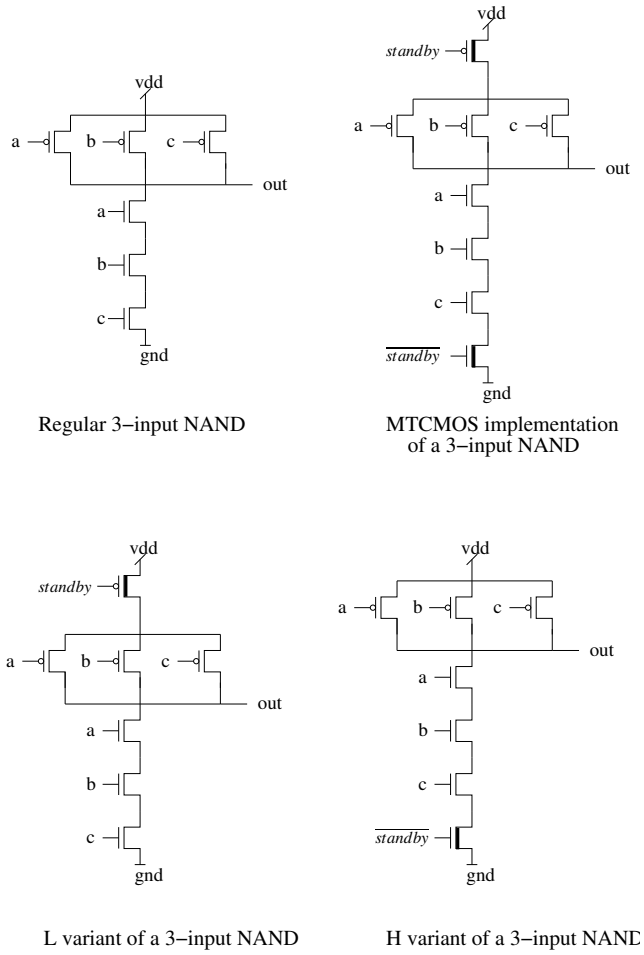


Fig. III.1. Transistor level description (NAND3 gate)

worst-case output delay penalty over all gate input transitions was no larger than 15% as compared to the regular standard-cell using low V_T transistors. In [9] too, the power supply gating transistors were sized such that their simulated delay penalties were no larger than 15%. Additionally, if the delay penalty desired is less than 15%, then the gate area overheads are quite significant. The sizes of the devices of the regular standard-cell were left unchanged in our MTCMOS and H/L cell variants.

If we were to modify the sizes of *all* devices of a gate (not just the header/footer

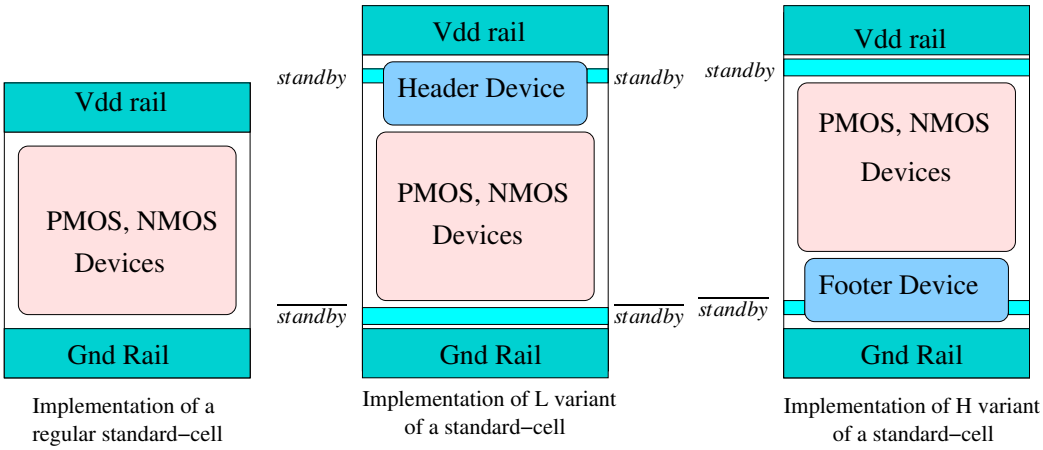


Fig. III.2. Layout floor-plan of HL gates

devices), we anticipate that our cell area overheads would be much smaller, and the cells could be faster for a given area overhead. However, this would involve layout of H/L cells from scratch. For the results reported here, we have made a decision to not modify the device sizes of the regular design in order to produce an approach which is easy to adopt in practice. With this choice, we have been able to generate the layouts of the H/L standard-cells by minimally modifying the layouts of the existing standard-cells.

Our H/L cell layouts are derived from the existing standard-cells by simply placing the VDD and GND rails of a cell further apart, in order to introduce just enough additional space to insert the header/footer devices. This is shown schematically in Figure III.2. Note that in the H and L variants of the regular standard-cell, the layout of the regular standard-cell devices (the region labeled "PMOS, NMOS Devices") is not modified. The *standby* and $\overline{standby}$ signals are routed by abutment, and run across the width of each H/L standard-cell. The header and footer transistors are implemented in a space-efficient zig-zag configuration as shown in the layout of Figure III.3. This also allows the header and footer device regions to be available for over-the-cell routing. In our simulations we assumed that the width of

the header and footer transistors to be equal to the center-line length of the poly shape. This is a common approximation used in circuit design. However for additional accuracy one can conceivably run existing commercial extraction tools to obtain an adjustment factor to account for the U-turns made in the poly shape. However, the adjustment factor is expected to be close to unity since there are only 2 U-turns in each H and L cell. Finally our HL cells have more pin landing sites, to enable ease of routing. In this manner, we were able to design H/L layout variants of each cell in an area-efficient manner.

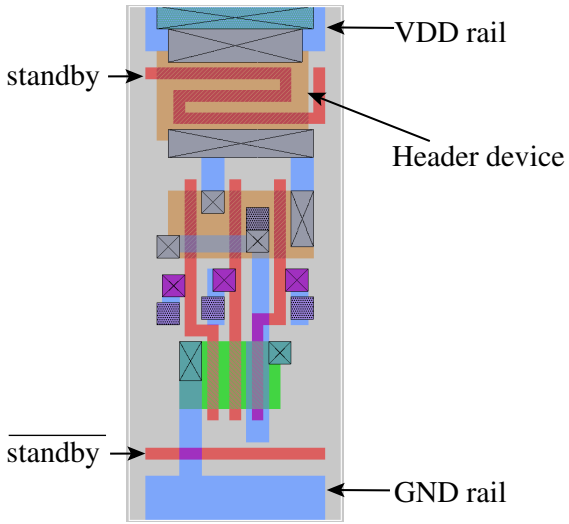


Fig. III.3. Layout of NAND3-L cell

III-D.1. Design Methodology

The overall design flow to implement a circuit using H/L standard-cells is very similar to a traditional standard-cell based design methodology. We first perform traditional mapping using regular standard-cells. After determining a set of primary input assignments for the standby mode of operation, we simulate the circuit with these assignments to determine the output of each gate. If the output of a gate is high, we replace it with the corresponding H

cell, and if it is low, with the L variant of the cell. Hence the decision of which cell variant to utilize for any given circuit can be made in time linear in the size of the circuit.

The schemes discussed in [28], [29] and [30] are similar to ours, but their authors do not mention that the leakage current in such a scheme is predictable. Also, in our HL methodology, the power supply gating devices are included within the standard-cell itself for simplicity. This ensures that we do not have to use ungated additional power supply rails which are required in the schemes of [28], [29] and [30]. We also perform detailed analysis of the delay-area trade-offs for an extensive set of benchmark circuits, which is discussed in Section III-E.

The determination of an optimal primary input assignment to utilize for the standby mode is actually a complex one. An Algebraic Decision Diagram [21] based framework can be used to solve this problem. In this approach, we would construct a decision diagram of a circuit topologically from primary inputs to primary outputs and assign each input vector a value of leakage based on the circuit state implied by that vector. Since the leakage current values for a cell differ for every input vector, we could potentially have an exponential number of decision nodes in the ADD. In order to develop an efficient algorithm for finding the best input vector, these different leakage values are in practice discretized. The granularity of this discretization would affect the optimality of the solution and the efficiency of the algorithm. Additionally, we could use a method of bounding the ADD leaf node values once a solution has been determined. For a scan enabled design, these primary inputs can be easily applied. If this is not the case, a phase-forcing circuit as discussed in [30] can be used to apply the required inputs to a combinational block.

III-D.2. Advantages and Disadvantages of the HL Approach

The advantages of the HL methodology are:

- By ensuring that each cell has a full-rail output value during standby operation, we make sure that the leakage of each standard-cell, and therefore the leakage of a standard-cell based design, are *precisely predictable*. Therefore our methodology avoids the unpredictability of leakage that results when using the MTCMOS style of design. This unpredictability occurs due to the fact that in MTCMOS, cell outputs, inputs and bulk voltages float to unknown values which are dependent on various processing and design factors.

- Since our inverting H/L cells utilize exactly one supply gating device (as opposed to two devices for MTCMOS), our cells exhibit better delay characteristics than MTCMOS for one output transition (the falling transition for L gates and rising transition for H gates). Though the authors of [9] mention that it possible to use only footer devices, their implementation uses both header and footer devices. Though using only a footer device will reduce the delay penalties, the leakage current increases as we show in Section III-E.

- For MTCMOS designs, memory elements would require clean power and ground supplies if they were to retain state during standby mode [9]. With the HL approach, the inputs to a combinational block are fixed in the standby mode. Hence the states of the memory elements that drive these inputs are also fixed. Therefore our technique can be applied to sequential elements as well (by using header devices when the leakage path is through the PMOS stack and using footer devices when the leakage path is through the NMOS stack). Alternatively, we could utilize the same flip-flop design as in [9]. In either case, the HL approach would *not* require special clean supplies to be routed to the flip-flop cell, resulting in lower area utilization for sequential designs.

- For many of the standard-cells, and particularly for larger cells which exhibit large

values of leakage, our H/L cells exhibit much lower leakage current. However, there are cells for which our cells exhibit comparable or greater leakage than MTCMOS as well. This is quantified in Section III-E.

- By implementing the header and footer devices in a layout-efficient manner, we ensure that the layout overhead of H/L standard-cells is minimized. Our choice of layout also allows the header and footer device regions to be free for over-the-cell routing.

The disadvantages of our approach are:

- The determination of the primary input assignments to utilize for the standby mode is a complex once. Although our current implementation makes this decision arbitrarily, it can be improved by applying the ideas described in Section III-D.1.

- Using the HL approach requires that the primary inputs to the circuit be driven to known values in the *standby* state. However if we assume that a combinational block of logic implemented using our approach is driven by flip-flops which are scan-enabled, then the required input vector can be simply scanned in before the circuit goes into the *standby* state. Alternatively, special circuitry (such as a NAND2 or a NOR2 gate with the standby signal as one of the inputs) could be added at the primary inputs.

- The experiments presented in this chapter can be improved if the technology mapping tools are modified. Assuming the primary input vector is predetermined and that we use a dynamic programming based technology mapper, the mapper would need to store the best match at any node as well as the logic state of that best match. For any new node that is being mapped, its logic state can therefore be determined, and so we would know whether to use a H or L cell for that mapping. In either case, we would

know what delay or area value to use for an optimum match at that node. In reality, the problems of technology mapping and the determination of an optimal primary input vector are coupled.

- Our method requires that the standby signals be routed to each cell. However, we have overcome this problem by designing the layout of H/L cells such that the routing of standby signal is performed by abutment, while also leaving free space for over-the-cell routing above the region where the standby signals are run.

III-E. Experimental Results

The standards cells we used were taken from the low-power standard-cell library of [31]. Our standard-cell library consisted of the following cells: INVA, INVB, NAND2A, NAND2B, NAND3, NAND4, NOR2, NOR3, NOR4, AND2, AND3, AND4, OR2, OR3, OR4, AOI21, AOI22, OAI21 and OAI22. The H and L variants of each of the standard-cells were created by modifying (adding high-V_T header and/or footer devices as required) the regular cells. The header and footer devices used in the HL variants as well as the MTCMOS cells were sized such that the worst-case cell delays were within 15% of the regular standard-cell worst-case delays. The sizes of the other transistors were not changed for reasons mentioned in Section III-D.

We used SPICE3f5 [32] for simulations of the standard-cells. The NMOS and PMOS model cards used were derived from the *bsim100* model cards [27]. The threshold voltages of the high-V_T transistors were 200mV greater than those of the regular devices. A supply voltage of 1.2V was assumed.

After performing the design, layout and characterization of individual cells, we compared the leakage, delay and area characteristics of the HL, MTCMOS and regular standard-cell based design methodologies for a set of circuits taken from the MCNC91 benchmark

suite.

In Figure III.4, we plot the range of leakage values for each MTCMOS cell against the range of leakage values obtained using the corresponding HL cell. For the HL cells, all possible input vectors were applied for each cell. This gave us the range of leakage values possible for the HL cells. Finding the range of leakage for the MTCMOS cells, is not as straightforward as finding the leakage for HL cells, since the inputs to the MTCMOS cell are not full-rail values during standby. For our experiments, we applied all possible voltage values from 0 to 1.2V, in steps of 0.2v, at each input of the MTCMOS cells and then found the minimum and maximum leakage currents. Note that in Figure III.4, we have also compared the range of leakage values for MTCMOS cells using only header sleep transistors and for MTCMOS cells using only footer sleep transistors. From Figure III.4, we find that the range of leakage values for the MTCMOS cells using both header and footer sleep transistors is much smaller than the range of leakage values when only one of the devices is used. Hence, from this point on, we use only the MTCMOS cells with both header and footer devices for comparisons with our H/L cells.

III-E.1. Comparison of Placed and Routed Circuits

A set of circuits from the MCNC91 benchmarks were implemented using all three design methodologies (regular standard-cell, HL and MTCMOS). Logic optimization and mapping were performed in the SIS [33] environment. The resulting leakage, area and delay numbers were compared. For circuits designed using H/L type cells, each primary input signal was assumed to be logic low in standby mode. The choice of selecting the H or L variant for each standard-cell was made as described in Section III-D.1.

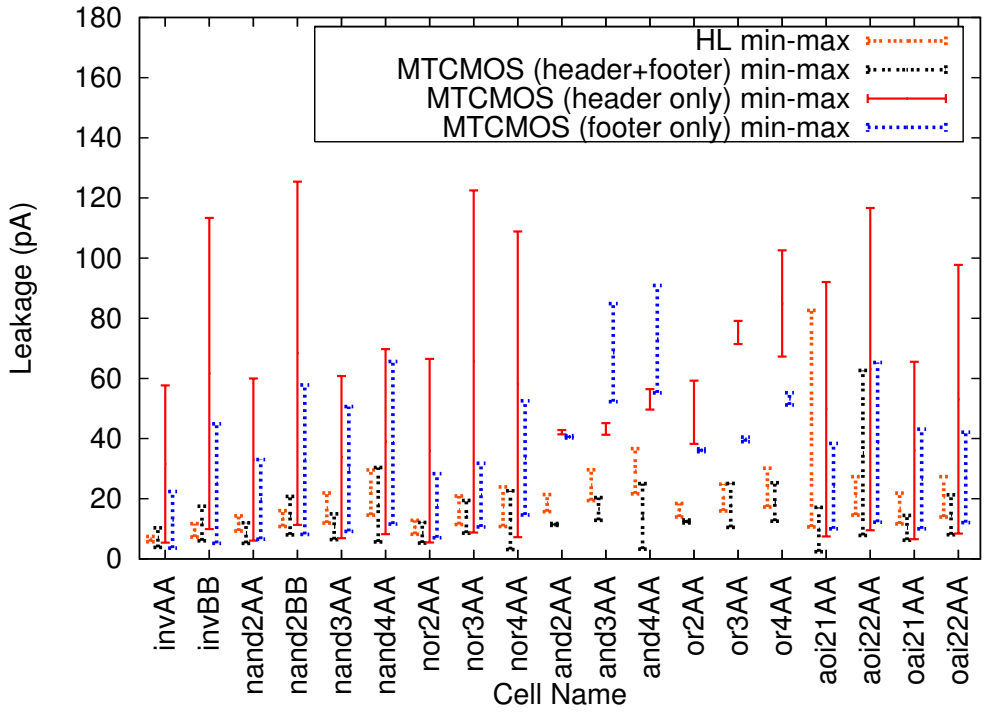


Fig. III.4. Plot of leakage range of HL versus MT method

III-E.1.a. Leakage Comparison

We first computed the leakage of each H/L cell based on the values of cell inputs implied by the applied primary input combination. Using this information, the leakage of the circuit mapped using the H/L gates was estimated by adding the leakage of the individual gates used. This is possible since the inputs to each gate in standby mode are known. We also ran SPICE on the mapped design, using the same primary input vector, to obtain a more accurate leakage estimate for the design. Figure III.5 is a scatter plot of the leakage values thus obtained, for all the circuits under consideration. From Figure III.5, we observe that for all the examples, the estimated leakage for the HL design and actual leakage obtained from SPICE are in very close agreement. This validates our claim that the *leakage for a HL design is precisely estimable* from the leakage values of each of its constituent gates.

Thus, if one were to design low-leakage circuitry using the HL methodology, the standby power consumption can be computed with great accuracy. This is in stark contrast with MTCMOS based designs.

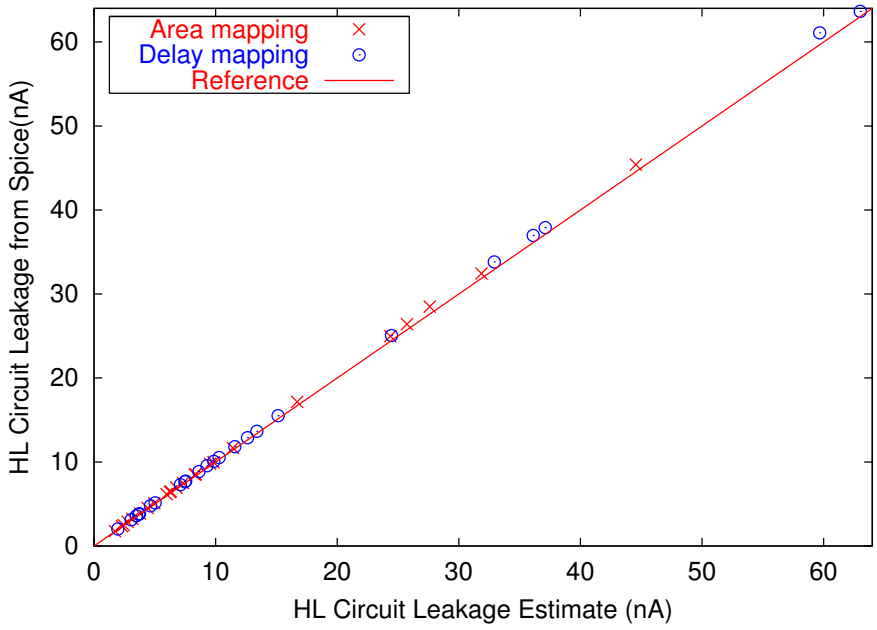


Fig. III.5. Leakage of HL-spice versus HL method over circuits

For the MTCMOS methodology, we determined the sum of the maximum and minimum leakage values of individual gates (these values were also previously estimated from SPICE simulations and reported in Figure III.4). The results are presented in Figures III.6 and III.7, and compared with the leakage of the HL methodology. In Figure III.6, the circuits were mapped for minimum area, while in Figure III.7, the circuits were mapped for minimum delay. In a mapped design, the inputs to the MTCMOS gates of the circuit would float in standby mode. Therefore the precise leakage value for the MTCMOS design is unpredictable, hence we used the maximum and minimum values of MTCMOS leakage as mentioned in the description for Figure III.4. In practice, the actual value of the leakage current for an MTCMOS circuit may well be greater than the maximum value as computed

above, based on the voltage values of the gate inputs and bulk nodes which float during standby.

Figures III.6 and III.7 indicate that the leakage of a design implemented using HL cells can be much smaller than the maximum leakage of a MTCMOS design. Note, that for the results presented here, we simply assumed that the primary inputs were set to logic 0. If we were to set the primary input vector to a state that minimized leakage, the leakage for our approach is expected to be even lower.

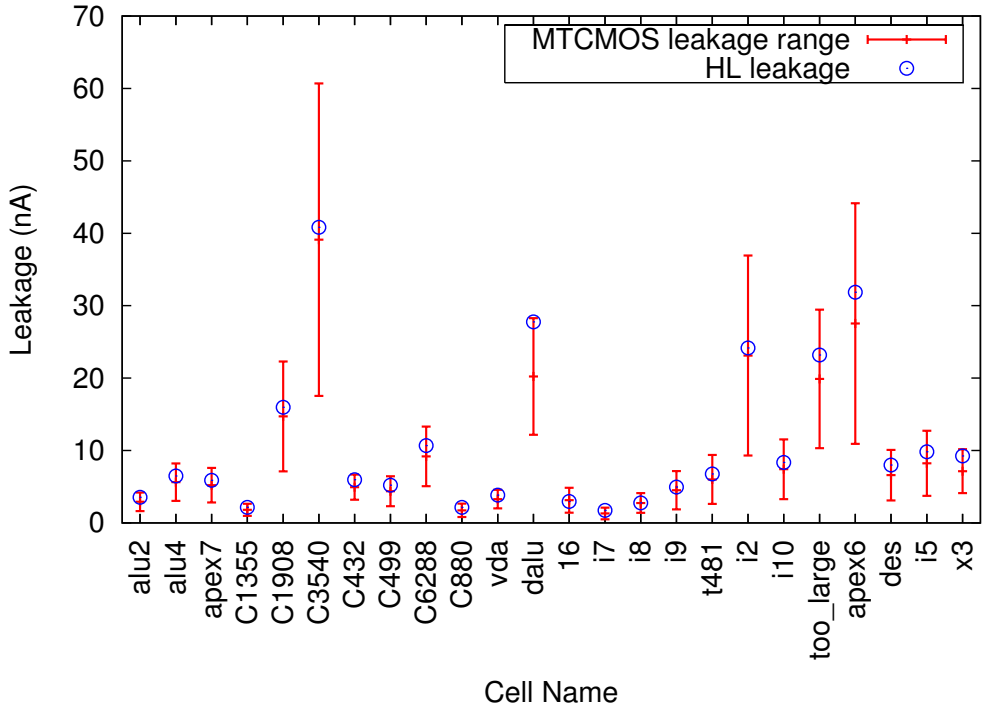


Fig. III.6. Leakage of HL versus MT (circuits mapped for min. area)

III-E.1.b. Delay Comparison

To compare the delay of the three techniques, we performed Exact Timing Analysis [34]. Given a mapped circuit, exact timing analysis returns the largest *sensitizable* delay for that

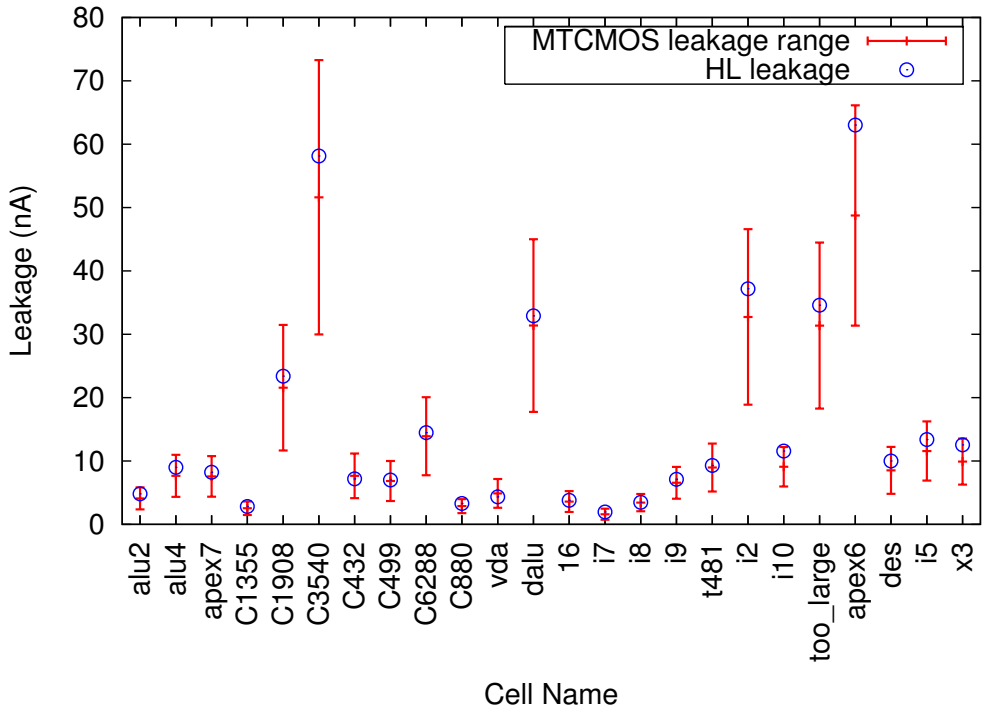


Fig. III.7. Leakage of HL versus MT (circuits mapped for min. delay)

circuit. As opposed to static timing analysis, exact timing eliminates false paths. We used the implementation of exact timing (the *sense* package which is implemented in SIS [33]) from the authors of [34].

To run *sense*, we generated a modified library description file for each of the three techniques. This file, in SIS's *genlib* format, describes the rising and falling delay from each input pin to the output pin for all gates in the library. Each such delay is a tuple consisting of a constant delay term and a load-dependent term. A standard-cell library characterization script was utilized to automatically generate this *genlib* file for all three design styles.

The results of *sense* are described in Table III.1 (for the case where mapping is done for delay minimization) and Table III.2 (for the case where mapping is done for area mini-

mization). For our benchmark suite of 24 examples, HL mapping exhibits a delay overhead of about 10% while MTCMOS exhibits an area overhead of 12.5%, compared to the regular method. As discussed earlier, the delay of the HL circuit is lower on account of the fact that only one transition of each gate is degraded in the process of modifying a gate for reduced leakage in the H/L approach. We also find that in two cases (apex7 and apex6 in Table III.1), the HL circuit actually has a small delay decrease. This is due to the fact that while adding a footer sleep device worsens the falling transition, the rising transition actually improves slightly. This is because the additional footer sleep device makes the path to ground more resistive and hence speeds up the rising transition. Similarly, falling transitions are improved slightly when a header sleep device is used. Hence in rare cases it is possible that a critical path gets sped up due to the addition of sleep transistors.

Table III.1. Delay (ps) comparison for all methods (delay mapping)

Example	Reg Delay	HL Delay	HL ovh.	MT Delay	MT ovh.
alu2	4146.65	4296.20	3.61	4546.15	9.63
alu4	5024.59	5135.15	2.20	5583.55	11.12
apex6	1660.15	1644.10	-0.97	1754.70	5.70
apex7	1959.00	1916.60	-2.16	2108.40	7.63
dalu	9270.03	10314.05	11.26	10494.15	13.21
des	14571.29	16690.05	14.54	16704.20	14.64
C1355	2567.91	2738.10	6.63	2922.80	13.82
C1908	3056.04	3403.45	11.37	3467.75	13.47
C3540	5756.18	6577.75	14.27	6537.05	13.57
C432	5309.39	5679.95	6.98	6015.25	13.29
C499	2289.99	2439.05	6.51	2586.20	12.93
C6288	13632.70	15528.65	13.91	15742.70	15.48
C880	2509.65	2853.90	13.72	2890.80	15.19
i2	610.55	652.70	6.90	665.95	9.07
i5	1136.75	1225.45	7.80	1232.35	8.41
i6	6698.08	7598.70	13.45	7610.40	13.62
i7	8074.18	9162.45	13.48	9174.15	13.62
i8	19027.58	21498.20	12.98	21799.45	14.57
i9	7370.84	8475.55	14.99	8503.00	15.36
i10	8479.30	8850.95	4.38	9680.85	14.17
t481	10040.29	11398.90	13.53	11374.05	13.28
too_large	4407.89	4809.00	9.10	4998.65	13.40
vda	3890.79	4329.05	11.26	4439.20	14.10
x3	2363.04	2653.60	12.30	2680.30	13.43
AVG			9.25%		12.61%

Table III.2. Delay (ps) comparison for all methods (area mapping)

Ckt.	Reg. Delay	HL Delay	HL ovh.	MT Delay	MT ovh
alu2	3971.00	4285.60	7.92	4474.70	12.68
alu4	6068.20	6797.55	12.02	6909.25	13.86
apex6	2248.85	2530.45	12.52	2500.20	11.18
apex7	1871.10	1925.60	2.91	2037.95	8.92
dalu	11868.45	12807.75	7.91	13198.00	11.20
des	19564.60	20593.90	5.26	22228.00	13.61
C1355	2952.80	3232.40	9.47	3383.60	14.59
C1908	4087.80	4689.80	14.73	4676.70	14.41
C3540	5730.85	6258.55	9.21	6528.40	13.92
C432	5220.30	5638.00	8.00	5893.10	12.89
C499	2723.60	3053.90	12.13	3117.60	14.47
C6288	11352.30	12912.65	13.74	13151.30	15.85
C880	2685.50	2963.30	10.34	2995.70	11.55
i2	703.00	763.60	8.62	787.60	12.03
i5	1154.70	1287.30	11.48	1270.80	10.05
i6	9182.30	10564.60	15.05	10409.20	13.36
i7	10549.85	11944.90	13.22	11781.10	11.67
i8	24974.05	28940.35	15.88	28675.30	14.82
i9	14746.35	16497.85	11.88	16576.30	12.41
i10	10335.00	11532.15	11.58	11664.95	12.87
t481	17192.70	19317.20	12.36	19092.50	11.05
too_large	4205.35	4650.85	10.59	4647.90	10.52
vda	5465.45	6140.05	12.34	6170.55	12.90
x3	3591.25	3986.60	11.01	3915.80	9.04
AVG			10.84%		12.49%

III-E.1.c. Area Comparison

We optimized and mapped our benchmark designs (for both minimum area and minimum delay) using SIS [33]. The circuits were then placed and routed using the Silicon Ensemble [35] tool set from Cadence Design Systems. Placement and routing was performed for both regular standard-cell and H/L cell based circuits, using 2, 3 and 4 metal routing layers. This gave us an accurate measure of the actual die area required to design circuits using these two methodologies. For the MTCMOS methodology, the header and footer "sleep" transistors are large devices which are shared by all the gates in a design. According to [8], one can exploit information about simultaneous transitions in a circuit to size sleep transistors efficiently. As stated earlier, this approach is not feasible for random logic circuits. Therefore, for MTCMOS circuits, we found the sum of the sizes of the MTCMOS headers and footers of the individual gates in the design. Based on this information, we estimated

the layout area overhead of MTCMOS. This overhead was then added to the area of the circuit implemented using regular cells. In an MTCMOS design, additional area needs to be devoted for routing an extra pair of power rails (see section III-D.2). This was neglected since our designs were combinational in nature. For sequential circuits, the MTCMOS overhead would therefore be higher. Tables III.3 and III.4 describe the area comparison results. The former table is obtained when technology mapping was performed for minimum delay, and the latter for minimum area. The tables show the total area (using a 0.1μ process) for regular standard-cell, HL cell and MTCMOS based circuits. The percentage area overhead for the HL and MTCMOS methods is also shown.

We note that on average, the HL design methodology exhibits a 11-30% area overhead compared to the regular design. However, the HL designs utilize on average up to 17% less area than the MTCMOS designs. As seen in Tables III.3 and III.4, the area overhead for MTCMOS does not decrease with increased metal layers, while the area overhead for HL does decrease. This is because the distributed nature of the sleep transistors in the HL scheme allows for more over-the-cell routing opportunities. The results validate the intuition that when more metal layers are used, the router can take advantage of over-the-cell routing and the area penalty for the HL methodology is reduced. For some examples, the HL designs exhibit a lower area than their regular counterparts. We conjecture that this is due to the fact that our HL cells are more router-friendly, with more over-the-cell routing space and also more pin landing sites.

III-F. Using Gate Length Biasing Instead of V_T Change

Recent research [36, 37] has suggested that *gate-length biasing* can be used alternative to multiple threshold voltage devices. Gate-length biasing is a technique by which small increases (5 to 10%) in the gate length are made, reducing leakage by as much as $2\times$. Gate-

Table III.3. Area (μ^2) comparison for all methods (delay mapping)

Ckt.	2-Layer						3-Layer						4-Layer					
	Reg. Area	HL Area	HL ovh.	MT Area	MT ovh.	HL-MTovh.	Reg. Area	HL Area	HL ovh.	MT Area	MT ovh.	HL-MTovh.	Reg. Area	HL Area	HL ovh.	MT Area	MT ovh.	HL-MTovh.
alu2	2480.04	3203.56	29.17	3422.48	38.00	-6.40	1713.96	2560.36	49.38	2656.40	54.99	-3.62	1713.96	2560.36	49.38	2656.40	54.99	-3.62
alu4	5184.00	6400.00	23.46	6964.54	34.35	-8.11	3249.00	4542.76	39.82	5029.54	54.80	-9.68	3576.04	4542.76	27.03	5356.58	49.79	-15.19
apex6	4928.04	5565.16	12.93	6740.40	36.78	-17.44	3624.04	4542.76	25.35	5436.40	50.01	-16.44	4070.44	4542.76	11.60	5882.80	44.52	-22.78
apex7	1156.00	1600.00	38.41	1756.09	51.91	-8.89	1089.00	1444.00	32.60	1689.09	55.10	-14.51	1089.00	1459.24	34.00	1689.09	55.10	-13.61
dalu	10816.00	14352.04	32.69	15509.27	43.39	-7.46	9101.16	12678.76	39.31	13794.43	51.57	-8.09	9101.16	12678.76	39.31	13794.43	51.57	-8.09
des	46397.16	51710.76	11.45	56678.33	22.16	-8.76	50086.44	29790.76	-40.52	60367.61	20.53	-50.65	48664.36	28425.96	-41.59	58945.53	21.13	-51.78
C1355	3203.56	4542.76	41.80	5059.68	57.94	-10.22	4070.44	4542.76	11.60	5926.56	45.60	-23.35	3672.36	4542.76	23.70	5528.48	50.54	-17.83
C1908	3387.24	4761.00	40.56	4912.76	45.04	-3.09	2851.56	3969.00	39.19	4377.08	53.50	-9.32	3249.00	3969.00	22.16	4774.52	46.95	-16.87
C3540	7744.00	10120.36	30.69	10871.04	40.38	-6.91	5806.44	8100.00	39.50	8933.48	53.85	-9.33	5806.44	7779.24	33.98	8933.48	53.85	-12.92
C432	1169.64	1747.24	49.38	1819.41	55.55	-3.97	1169.64	1681.00	43.72	1819.41	55.55	-7.61	1197.16	1681.00	40.42	1846.93	54.28	-8.98
C499	2134.44	3069.16	43.79	3252.29	52.37	-5.63	1936.00	2704.00	39.67	3053.85	57.74	-11.46	3624.04	2704.00	-25.39	4741.89	30.85	-42.98
C6288	13041.64	17476.84	34.01	21098.58	61.78	-17.17	11620.84	16952.04	45.88	19677.78	69.33	-13.85	11620.84	16952.04	45.88	19677.78	69.33	-13.85
C880	1814.76	2480.04	36.66	2586.02	42.50	-4.10	1444.00	2134.44	47.81	2215.26	53.41	-3.65	1428.84	2134.44	49.38	2200.10	53.98	-2.98
i2	1024.00	1398.76	36.60	1358.97	32.71	2.93	817.96	1142.44	39.67	1152.93	40.95	-0.91	817.96	1142.44	39.67	1152.93	40.95	-0.91
i5	1918.44	2560.36	33.46	2676.54	39.52	-4.34	3249.00	2134.44	-34.30	4007.10	23.33	-46.73	2916.00	2116.00	-27.43	3674.10	26.00	-42.41
i6	3576.04	4705.96	31.60	4999.98	39.82	-5.88	4070.44	3969.00	-2.49	5494.38	34.98	-27.76	4070.44	3969.00	-2.49	5494.38	34.98	-27.76
i7	6177.96	6115.24	-1.02	8054.29	30.37	-24.07	4070.44	5212.84	28.07	5946.77	46.10	-12.34	4070.44	5212.84	28.07	5946.77	46.10	-12.34
i8	20449.00	26830.44	31.21	27105.78	32.55	-1.02	21609.00	20449.00	-5.37	28265.78	30.81	-27.65	21609.00	20449.00	-5.37	28265.78	30.81	-27.65
i9	5184.00	6561.00	26.56	6824.72	31.65	-3.86	4435.56	5745.64	29.54	6076.28	36.99	-5.44	4019.56	5745.64	42.94	5660.28	40.82	1.51
i10	28968.04	30765.16	6.20	35796.49	23.57	-14.06	22560.04	18117.16	-19.69	29388.49	30.27	-38.35	24649.00	18117.16	-26.50	31477.45	27.70	-42.44
t481	24964.00	33489.00	34.15	33259.85	33.23	0.69	20334.76	29104.36	43.13	28630.61	40.80	1.65	20334.76	29104.36	43.13	28630.61	40.80	1.65
too_large	5685.16	7396.00	30.09	7456.22	31.15	-0.81	3769.96	5212.84	38.27	5541.02	46.98	-5.92	3769.96	5270.76	39.81	5541.02	46.98	-4.88
vda	7992.36	10000.00	25.12	10111.07	26.51	-1.10	4900.00	6822.76	39.24	7018.71	43.24	-2.79	4928.04	6822.76	38.45	7046.75	42.99	-3.18
x3	6304.36	7499.56	18.96	8285.65	31.43	-9.49	4489.00	5745.64	27.99	6470.29	44.14	-11.20	4928.04	5745.64	16.59	6909.33	40.20	-16.84
AVG			29.08		38.94	-7.05			24.89		45.61	-14.96			20.70		43.97	-16.95

Table III.4. Area (μ^2) comparison for all methods (area mapping)

Ckt.	2-Layer						3-Layer						4-Layer					
	Reg. Area	HL Area	HL ovh.	MT Area	MT ovh.	HL-MTovh.	Reg. Area	HL Area	HL ovh.	MT Area	MT ovh.	HL-MTovh.	Reg. Area	HL Area	HL ovh.	MT Area	MT ovh.	HL-MTovh.
alu2	2097.64	2560.36	22.06	2626.41	25.21	-2.51	1398.76	1764.00	26.11	1927.53	37.80	-8.48	1296.00	1764.00	36.11	1824.77	40.80	-3.33
alu4	4356.00	5685.16	30.51	5343.56	22.67	6.39	2894.44	3481.00	20.27	3882.00	34.12	-10.33	2601.00	3528.36	35.65	3588.56	37.97	-1.68
apex6	3721.00	4435.56	19.20	4667.28	25.43	-4.96	4070.44	3136.00	-22.96	5016.72	23.25	-37.49	4542.76	3113.64	-31.46	5489.04	20.83	-43.28
apex7	912.04	1296.00	42.10	1257.38	37.86	3.07	795.24	1142.44	43.66	1140.58	43.43	0.16	795.24	1142.44	43.66	1140.58	43.43	0.16
C1355	2323.24	3433.96	47.81	3324.24	43.09	3.30	2894.44	2981.16	3.00	3895.44	34.58	-23.47	2209.00	2981.16	34.96	3210.00	45.31	-7.13
C1908	2601.00	3624.04	39.33	3417.87	31.41	6.03	1918.44	2601.00	35.58	2735.31	42.58	-4.91	2894.44	2601.00	-10.14	3711.31	28.22	-29.92
C3540	6241.00	8281.00	32.69	7844.76	25.70	5.56	4225.00	5745.64	35.99	5828.76	37.96	-1.43	4489.00	5745.64	27.99	6092.76	35.73	-5.70
C432	817.96	1156.00	41.33	1116.43	36.49	3.54	729.00	1011.24	38.72	1027.47	40.94	-1.58	729.00	1011.24	38.72	1027.47	40.94	-1.58
C499	1764.00	2480.04	40.59	2381.99	35.03	4.12	1521.00	2134.44	40.33	2138.99	40.63	-0.21	1521.00	2135.36	40.39	2138.99	40.63	-0.17
C6288	10774.44	15525.16	44.09	15035.06	39.54	3.26	10281.96	12100.00	17.68	14542.58	41.44	-16.80	9025.00	12056.04	33.58	13285.62	47.21	-9.25
C880	1369.00	1989.16	45.30	1859.69	35.84	6.96	1197.16	1648.36	37.69	1687.85	40.99	-2.34	1197.16	1648.36	37.69	1687.85	40.99	-2.34
dalu	9254.44	11793.96	27.44	11834.39	27.88	-0.34	6304.36	8353.96	32.51	8884.31	40.92	-5.97	6304.36	8353.96	32.51	8884.31	40.92	-5.97
des	45710.44	47089.00	3.02	51786.20	13.29	-9.07	51529.00	22801.00	-55.75	57604.76	11.79	-60.42	51892.84	22560.04	-56.53	57968.60	11.71	-61.08
i2	772.84	1142.44	47.82	1041.92	34.82	9.65	817.96	985.96	20.54	1087.04	32.90	-9.30	817.96	985.96	20.54	1087.04	32.90	-9.30
i5	1681.00	2246.76	33.66	2210.91	31.52	1.62	1197.16	1600.00	33.65	1727.07	44.26	-7.36	1197.16	1600.00	33.65	1727.07	44.26	-7.36
i6	3433.96	3069.16	-10.62	4172.76	21.51	-26.45	3624.04	2560.36	-29.35	4362.84	20.39	-41.31	4070.44	2560.36	-37.10	4809.24	18.15	-46.76
i7	4928.04	5184.00	5.19	5868.12	19.08	-11.66	4070.44	3203.56	-21.30	5010.52	23.10	-36.06	3624.04	3203.56	-11.60	4564.12	25.94	-29.81
i8	17902.44	19656.04	9.80	21382.48	19.44	-8.07	18988.84	12769.00	-32.76	22468.88	18.33	-43.17	18769.00	12588.84	-32.93	22249.04	18.54	-43.42
i9	4329.64	4928.04	13.82	5415.02	25.07	-8.99	4070.44	3969.00	-2.49	5155.82	26.67	-23.02	4070.44	3969.00	-2.49	5155.82	26.67	-23.02
i10	28968.04	29584.00	2.13	32519.46	12.26	-9.03	19656.04	14448.04	-26.50	23207.46	18.07	-37.74	21609.00	13409.64	-37.94	25160.42	16.43	-46.70
t481	20107.24	25027.24	24.47	24616.10	22.42	1.67	12321.00	17056.36	38.43	16829.86	36.59	1.35	12321.00	17056.36	38.43	16829.86	36.59	1.35
too_large	5155.24	6432.04	24.77	6232.95	20.91	3.19	3069.16	4096.00	33.46	4146.87	35.11	-1.23	3249.00	4019.56	23.72	4326.71	33.17	-7.10
vda	7022.44	8427.24	20.00	8139.24	15.90	3.54	4277.16	5387.56	25.96	5393.96	26.11	-0.12	4225.00	5329.00	26.13	5341.80	26.43	-0.24
x3	5041.00	6822.76	35.35	6400.07	26.96	6.60	4984.36	4542.76	-8.86	6343.43	27.27	-28.39	5929.00	4542.76	-23.38	7288.07	22.92	-37.67
AVG			26.74		27.06	-0.52			11.82		32.47	-16.65			10.84		32.36	-17.55

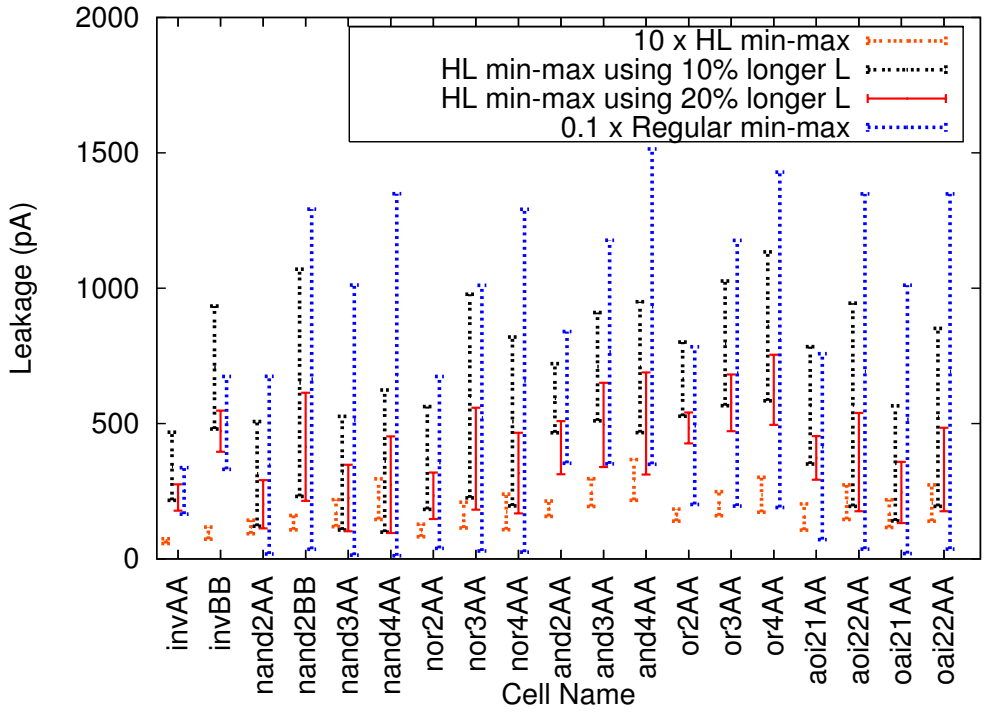


Fig. III.8. Plot of leakage range of H/L cells, H/L cells with gate length bias and regular cells

length biasing does not require additional lithography masks and is hence inexpensive to implement. We replaced the high-V_T devices in the H/L cells with devices with longer channel length (and low V_T) in an effort to see how this would affect the delay and leakage of the H/L cells. We tried gate lengths that were 10% higher and 20% higher than nominal (100nm). The minimum and maximum leakages obtained for each of the cells are shown in Figure III.8.

Note that in Figure III.8, the leakage of the regular H/L cells (that use high-V_T header or footer transistors) have been multiplied by a factor of 10, while the leakage of the regular cells (without any sleep transistors) have been divided by a factor of 10. As Figure III.8 shows, the new H/L cells that use gate-length biasing (instead of high-V_T devices) for the sleep transistors have a leakage that is between 1 and 2 orders of magnitude smaller than

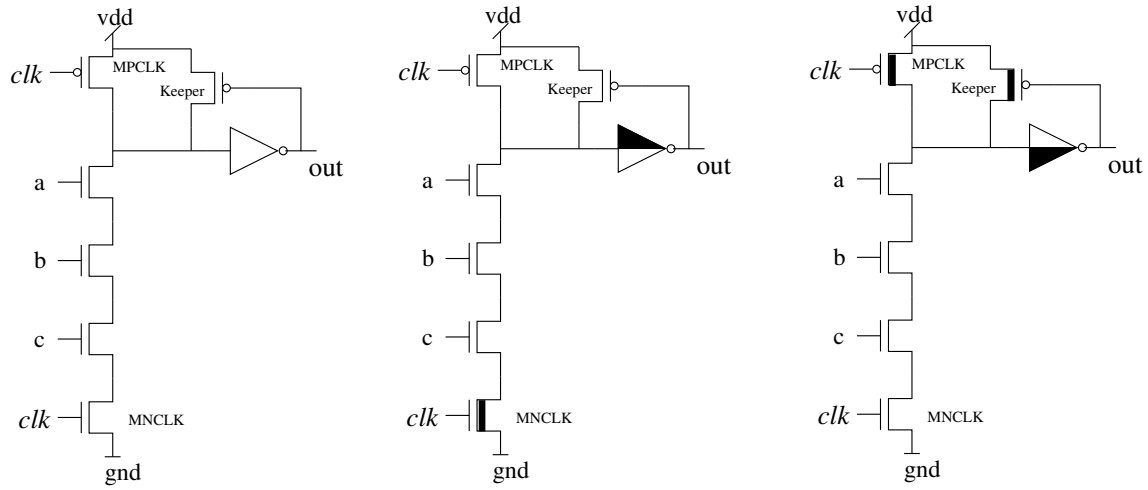
the leakage of regular cells. However, their leakage is between 1 and 2 orders of magnitude *greater* than the leakage of the regular H/L cells. We also simulated the new H/L cells and compared their delay impact. We found that the delay difference between the new H/L cells and the regular H/L cells was negligible. When compared to the regular H/L cells, the new H/L cells that used a gate-length biasing of 10% had between 1% to 3% smaller delay. For the new H/L cells that used a gate-length biasing of 20%, the delays were about the same as the regular H/L cells. Hence, we find that for the HL methodology, using high-V_T devices is more effective than using longer channel length devices since it gives a greater leakage reduction with a similar delay penalty. However, in case the cost associated with the additional threshold implant masks is to be avoided, one could use the H/L approach with gate-length biasing to obtain a leakage improvement over regular standard-cells.

III-G. Leakage Reduction in Domino logic

In this section we explore how leakage power reduction is achieved in dynamic cells. Specifically we focus on domino logic cells due to their widespread popularity.

In standby mode, domino logic gates can either be in the *precharge* or *evaluate* state. In either case, if dual V_T technologies are used, devices which are turned *off* (devices in the cut-off mode of operation) in standby mode are implemented with high V_T. This can typically reduce leakage currents by about 2 orders of magnitude.

Figure III.9 illustrates the low leakage alternatives for a domino logic AND3 gate. Figure III.9(a) is a traditional domino AND3 gate. Figure III.9(b) illustrates the design of an AND3 domino logic gate which, in standby mode, is held in the precharge state (*clk* signal is logic-0). In this mode the PMOS pull-up device (MPCLK) is turned *on* and the NMOS pull-down stack is turned off. In the output inverter, the PMOS device is turned off and the NMOS device is turned on. The advantage in this method is that we have at



a) Regular implementation of a domino 3−input AND

b) 3−input domino AND precharged in standby mode

c) 3−input domino AND evaluated in standby mode

Fig. III.9. Transistor level description (domino AND3 gate)

least 2 devices turned *off* in series in the NMOS stacks thus minimizing the leakage current. The footer device (MNCLK) and the PMOS device in the output inverter (illustrated by a dark triangle on the top part of the output inverter) are made high V_T to reduce leakage current further. However, both these devices are in the critical evaluate path of the domino logic gate, so the delay of the gate is increased when these devices are made high V_T. Therefore, these devices have to be up-sized to compensate for the increased delay. Rather than increasing the size of the footer device (MNCLK) alone, increasing the size of the rest of the devices in the NMOS stack results in smaller area penalties for the same delay.

Alternatively, the domino logic gate could be held in the *evaluate* state (NMOS stack turned *on*) during standby. In [8], the authors suggest such a method for a clock delayed domino logic scheme. An AND3 domino logic gate which is held in the *evaluate* state during standby is shown in Figure III.9(c). In standby mode the *clk* line is pulled high, thus turning off the PMOS pull-up device (MPCLK) and the NMOS in the output inverter.

These devices are implemented with high V_T devices to keep the leakage current low. The *keeper* device is also made a high V_T device. The advantage of this scheme is that only the devices in the precharge path are made high V_T and any delay increase is exhibited only in that path. We found that the delay in the evaluate mode is in fact decreased slightly due to reduced leakage contention from the high V_T PMOS device (MPCLK).

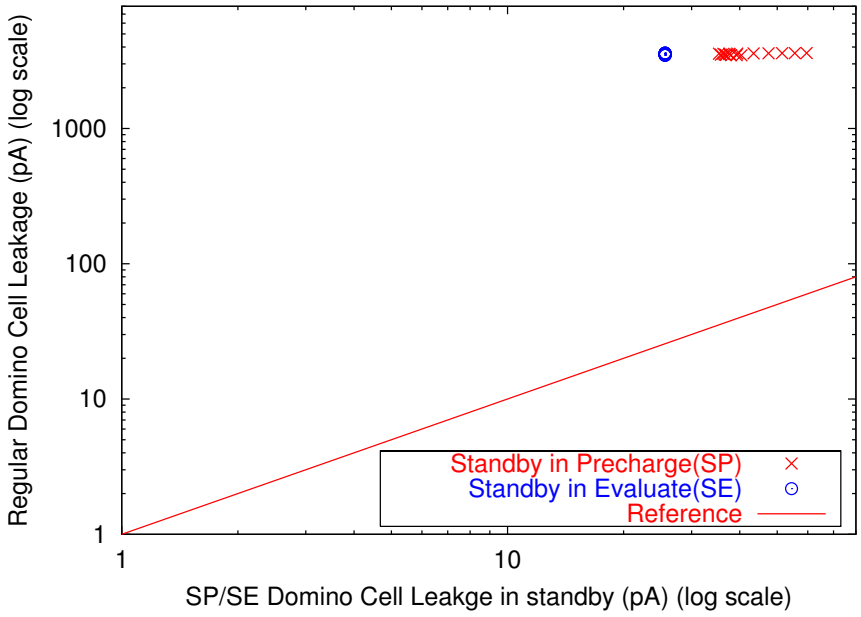


Fig. III.10. Leakage of SE/SP versus regular domino cells

A comparison of the leakages of the different schemes for a library of cells (cells compared were AND2 AND3 AND4 AND5 AND6 AOI21 AOI22 OAI21 OAI22 OR2 OR3 OR4 OR5 OR6 OR7 and OR8) is shown in Figure III.10. The scheme in which cells are held in *precharge* during standby is referred to as SP, and SE denotes the scheme in which cells are held in *evaluate* state during standby. In a regular domino logic gate, all devices are low V_T devices. Devices in the evaluate path of SP gates were up-sized such that the gate delay (in the evaluation phase) was made equal to the regular domino logic gate. As can be seen from Figure III.10, the leakage of SP and SE cells is dramatically lower (by

Table III.5. Leakage comparison SE vs SP

Ckt.	SP Leakage(pA)	SE Leakage(pA)	Ovh (%)
alu2	17516.82	12290.93	-29.83
alu4	36614.08	25913.37	-29.23
apex6	21543.77	15261.24	-29.16
apex7	7266.66	5146.83	-29.17
dalu	82461.74	58253.88	-29.36
des	166870.79	112001.09	-32.88
C1355	29497.98	21099.43	-28.47
C1908	27958.99	19588.67	-29.94
C3540	60278.10	41968.40	-30.38
C432	9184.09	6401.53	-30.30
C499	23250.06	16592.75	-28.63
C6288	165015.41	118914.73	-27.94
C880	14452.54	10140.02	-29.84
i2	3430.88	2048.49	-40.29
i5	7455.75	5095.62	-31.66
i6	10397.28	6913.66	-33.51
i7	12963.03	8501.24	-34.42
i8	52224.02	34542.67	-33.86
i9	16348.30	10626.55	-35.00
i10	101053.51	69443.76	-31.28
t481	47207.10	30164.03	-36.10
too_large	17053.89	11650.78	-31.68
vda	19747.06	12777.45	-35.29
x3	23492.55	16157.45	-31.22
Avg			-31.64

about 2 orders of magnitude) than that of regular domino logic cells (for the same delay). Also it can be seen that the leakage for the SE scheme does not change much across the different gates. This is because the leaking devices, the PMOS pull-up device (MPCLK) and the NMOS device in the output inverter, are of the same size for all gates. Leakage for SE cells was determined to be lower than the SP cells, as illustrated in Figure III.10. This is because the high V_T devices in the SP cells had to be up-sized in order to avoid increased gate delays.

We also compared leakages of the SE and SP schemes for a set of circuits. The results are shown in Table III.5. The leakage for the SE scheme is on average 31% lower than the SP scheme.

From the above, it is clear that using SE domino logic gates is a better option from a delay, leakage and cell area standpoint (as compared to SP domino logic gates). For an

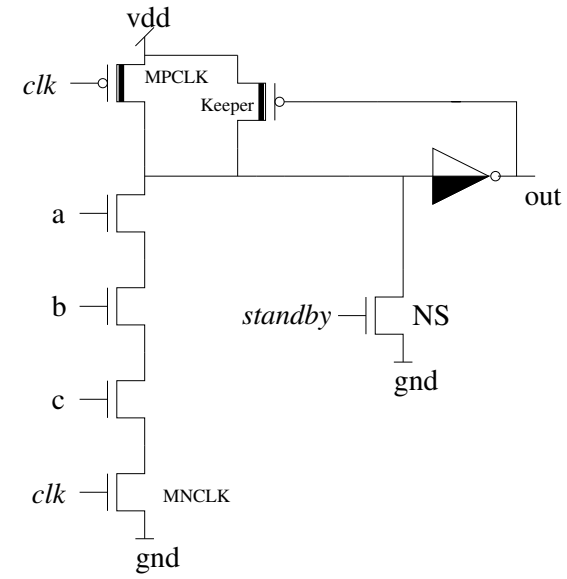


Fig. III.11. Transistor level description of first SE domino gate in a chain

SE domino logic gate, we need to ensure that all inputs of the gate are at logic-1 during standby mode. This can be done by gating the inputs of the first gate in a chain of domino logic cells. However, this will increase the delay of the gate during normal operation. The authors of [38] suggest a simple and elegant alternative. In this approach, an NMOS switch NS (as shown in Figure III.11) is used to pull down the dynamic node of the first gate in the chain. This switch is controlled by the standby signal. The only disadvantage of this method is that an additional standby signal is needed for the first gates in a chain of domino logic cells.

III-H. Chapter Summary

In this chapter, we have described low-leakage standard-cell based ASIC design methodologies for both static CMOS and domino logic. The major contribution is the development of a new methodology for low-leakage static CMOS designs, which we call the "HL" methodology. This "HL" methodology is based on ensuring that during standby operation, the supply voltage is applied across more than one *off* device and there is at least one *off* device with a high V_T in the leakage path. For each standard-cell in a library, we design two variants, the "H" and the "L" variant.

Our HL cells exhibit low leakage currents as do MTCMOS gates, but with the advantage that leakage currents in our methodology can be precisely estimated (unlike MTCMOS). We compared the two techniques using 24 placed-and-routed designs. We have shown that our methodology has a lower delay than MTCMOS, which is expected since our HL cells exhibit a delay degradation for only one output transition. Our HL designs exhibit *predictable* leakage values which are much lower than the maximum leakage for MTCMOS designs. Since leakage in MTCMOS designs is not precisely controllable, this is a significant improvement. Further, our HL designs exhibit an area overhead of approximately 21%-29% and 11%-27% over regular designs (for delay-optimal and area-optimal mapping respectively), and an area saving of up to 17% over MTCMOS designs. The HL methodology utilizes existing mapping and place/route tools, and handles memory elements without additional routing overhead (unlike MTCMOS). We also explored the use of header and footer devices with long channel length instead of high-V_T devices in the H/L cells. We found that a higher V_T device was more effective. It gave a smaller leakage with a similar delay penalty.

With the downward scaling of VDD in future technologies, the threshold voltages of both the high V_T and low V_T devices in the HL methodology will have to scale down as

well, if circuit delays are to be kept within reasonable limits. However, this could increase the leakage current. So if leakage current is the overriding concern, the V_T of the high V_T power supply gating devices should not be scaled down. Though this may cause an increase in delays, this increase is in only one transition for each gate unlike traditional MTCMOS. Hence the problems due to scaling of VDD in future technologies are similar for both MTCMOS and HL methodologies, but are worse for MTCMOS.

CHAPTER IV

SIMULTANEOUS INPUT VECTOR CONTROL AND CIRCUIT MODIFICATION

IV-A. Chapter Overview

Most approaches to reducing leakage have an associated performance penalty. One of the techniques used to minimize leakage is the technique of *parking* a circuit in its minimum leakage state. This technique involves very little or no circuit modification and does not require additional power supplies. A combinational circuit is *parked* in a particular state by driving the primary inputs of the circuit to a particular value during standby. This value can be scanned in via scab-enable flip-flops or forced using MUXes (with the standby/sleep signal used as a select signal for the MUX). This technique for leakage reduction is frequently referred to as *input vector control*. In this chapter we propose an approach that modifies and improves this technique to substantially to achieve control over the leakage of a circuit at a finer granularity. We present an approach which minimizes leakage by *simultaneously* modifying the circuit while deriving the input vector that minimizes leakage. In our approach, we selectively modify a gate so that its output (in sleep mode) is in a state which helps minimize the leakage of other gates in its transitive fanout. Gate replacement is performed in a slack-aware manner, to minimize the resulting delay penalty. One of the major advantages of our technique is that we achieve a significant reduction in leakage without increasing the delay of the circuit.

The leakage reduction technique discussed in this chapter is orthogonal to other circuit level leakage reduction approaches such as MTCMOS and others that statically (or dynamically) change the V_T of the devices.

IV-B. The Intuition Behind Our Approach

Table IV.1 shows the leakage of a NAND3 gate for all possible input vectors to the gate. The leakage values shown are from a SPICE [32] simulation using the 0.1μ BPTM [27] models at 1.2V.

Table IV.1. Leakage of a NAND3 gate

Input	Leakage(A)
000	1.37e-10
001	2.70e-10
010	2.70e-10
011	4.96e-09
100	2.62e-10
101	2.68e-09
110	2.51e-09
111	1.01e-08

As can be seen from Table IV.1, setting a gate in its minimal leakage state (000 in the case of the NAND3 gate) can reduce leakage by about 2 orders of magnitude. This leakage reduction is attributed to the *stack effect*, according to which having as many *off* transistors in series as possible minimizes leakage. While it is desirable to set every gate in a circuit to its minimal leakage state, it may not be possible to do so due to the logical inter-dependencies of the inputs of the gates. Even if the individual gates have a wide range of leakage values, this does not mean that a multi-level circuit that uses these gates will have a wide range of leakage values as well. For example if a NAND3 gate and a NOR3 gate in a circuit share inputs, the leakage of the NAND3 is minimum when all the inputs are set to logic 0, but to get the NOR3 gate into its minimum leakage state requires all the inputs to be set to logic 1. Due to such constraints, we are limited in terms of the leakage reduction that we can achieve by using just vector control at the primary inputs. In order to exploit the stack effect better, we need a technique that offers more freedom in setting the inputs at each gate. Herein lies the key contribution of this chapter.

In practice gate leakage currents can also contribute to the total leakage of a gate.

However, the contribution of gate leakage only affects the table of leakage values for each input vector for a gate. Our algorithm is agnostic to this and only requires a reliable estimate of leakage currents of a gate for different input vectors, and hence it can account for gate leakage as well.

IV-C. Related Previous Work

In an effort to exploit input vector control to minimize leakage, the problem of finding the minimum leakage sleep vector for a combinational CMOS gate-level circuit has received some attention recently. There are several heuristics([15, 16, 18, 17, 20, 25, 24, 22, 21]) that have been proposed to find the minimum leakage sleep vector. Some of these have been discussed in Chapter II. While these heuristics attempt to find the minimum leakage vector assuming that only the primary inputs of a combinational circuit can be controlled, *we focus on circuit modifications as well,to ensure that we are not restricted to the primary inputs alone, to control leakage*.

Traditionally, input vector control has involved using MUXes or scan-chains to control the primary input values of a circuit during *standby*. We extend this idea further and give ourselves the freedom to set the inputs of individual gates in a circuit. We modify the circuit such that we are not restricted to controlling just the primary inputs, but can also control the internal nodes of a circuit. While the idea of adding control points is similar to what is expressed in [39, 40], we allow a greater degree of freedom. In [39, 40], the authors insert either AND or OR gates to set the logic value of a particular line during *standby*. We, on the other hand, allow one input going to 2 or more different gates to be split (using pass-gate MUXes), so that each fanout can be set to different values during standby. This provides significantly more opportunities to control internal nodes and minimize leakage. Also in [39, 40], the authors use a SAT based algorithm to find control points and to min-

imize leakage. The accuracy of the algorithm is dependent on the number of quantization levels of leakage values. However, with a higher number of quantization levels the runtime also increases. The algorithm we use has significantly lower complexity, and involves a single linear-time traversal of the circuit. In [41], a technique is presented which involves gate replacement. However, in [41] a gate G is replaced by a different gate G' to only reduce the leakage of gate G, *but not to control other internal circuit nodes*. The authors of [42] improve on the implementation of [41] in terms of both leakage improvement and runtime of the gate replacement algorithm.

Previous approaches to minimize leakage through vector control and gate replacement [39, 40, 41, 42], have an associated delay penalty to get a reasonable leakage reduction. In our approach, we get a significant leakage reduction (as shown in Section IV-E) with *no delay penalty*.

IV-D. Our Approach

The algorithm we use in this chapter to minimize leakage and find control points in the circuit is designed to make sure that we don't ever get a negative slack. We have a built-in static timer that allows us to test if a gate violates timing.

One of the sources of our flexibility in controlling internal nodes of a circuit stems from the fact that we create several different variants of each gate in the library. While it may be argued that the creation of different variants of a cell can be time consuming and expensive, it should be noted that this step is done up-front and only once. An example of the different variants is shown in Figure IV.1. In the *snglmx* type of variant, a MUX is placed at the output of a regular gate. There are two type of *snglmx* gates, $snglmx_0$ and $snglmx_1$. The $snglmx_0$ gates have a weak pull-down device at the output of the MUX. A $snglmx_0$ variant is used as a replacement for a gate when the output of a gate G is logic 1 in

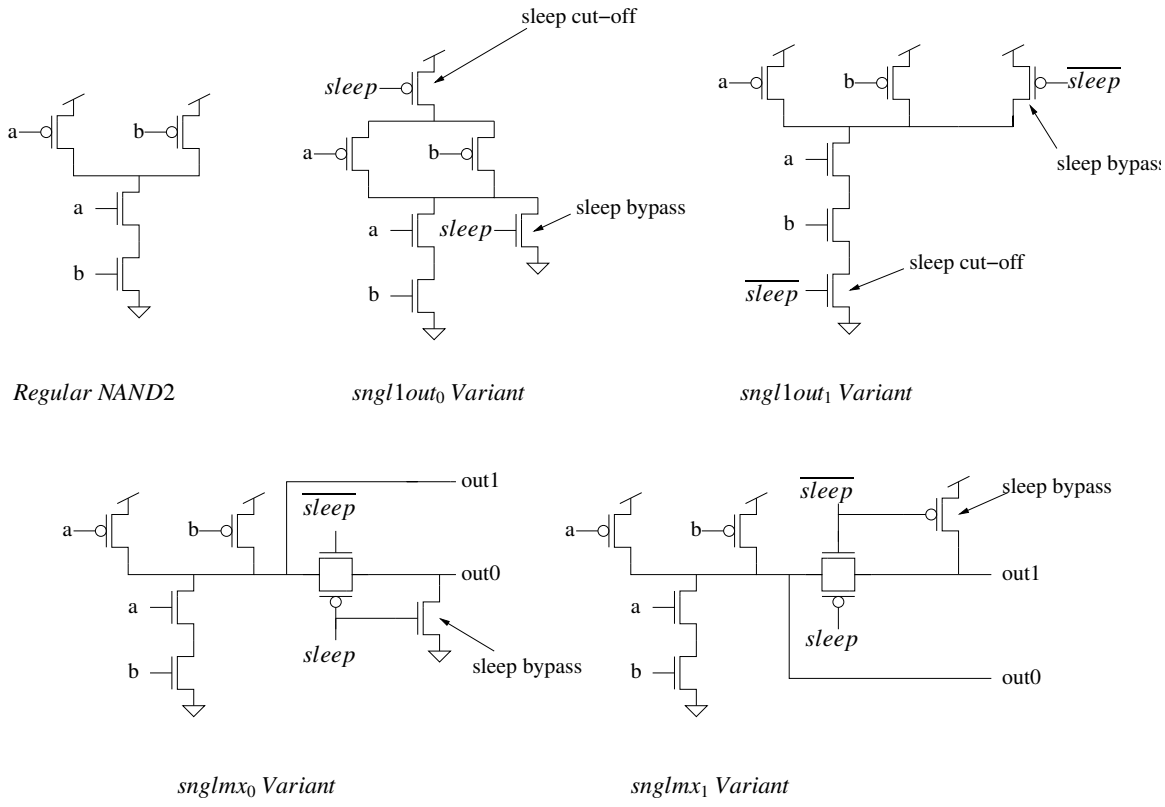


Fig. IV.1. Some variants of a NAND2 gate

standby, but some gates in the fanout of G require a logic 0 to get into a low leakage state. Similarly, $snglmx_1$ gates have a weak pull-up device at the output of the MUX. A $snglmx_1$ variant is used when the output of a gate G is logic 0 in standby, but some gates in its fanout require a logic 1 to get into a low leakage state. Note that the *snglmx* type of variants are dual output gates and hence offer the most flexibility by 'splitting' internal signals.

There can be situations when all the gates in the fanout of the gate in question need a value that is complementary to what is generated at the output of a gate in standby. For such cases we have a type of variant called the *sngl*1*out* variant. This type of variant has only 1 output and is similar to the structure discussed in [40]. We define 2 types of

*sngl*1*out* variants, *sngl*1out_0 and *sngl*1out_1. The *sngl*1out_0 uses a PMOS sleep transistor to cut-off the PMOS stack of the gate (labeled as sleep cut-off in Figure IV.1) and a weak NMOS pull-down device (labeled as sleep bypass in Figure IV.1) to pull down the output. This variant is used when the output of a gate is high in the standby state, while all the gates in the fanout require a logic low value to get into a low leakage state. Similarly the *sngl*1out_1 uses a NMOS sleep transistor to cut-off the NMOS stack of the gate and a weak PMOS pull-up device to pull up the output. This variant is used when the output of a gate is low in the standby state, while the gates in the fanout require a logic high value to get into a low leakage state. Note that while the *snglmx* type of variant worsens both output rise and output fall delays, the *sngl*1*out* worsens delay for either only the rise or only the fall transition and can actually speed up the opposite transition. *In [40], the authors take advantage of this fact and assume the delay of such a gate to be the average of the rise and fall delays. This assumption can lead to inaccuracies in the timing analysis.* In our approach, we account for the rise and fall delays separately.

Due to the introduction of sleep devices, the delay of the *sngl*1*out* gates is larger than the regular cells (for one transition). Similarly, the *snglmx* variants also suffer a delay due to the pass gate MUX at the output. Since we have output timing constraints, this delay limits the flexibility of the gate replacement algorithm. To enhance the flexibility of the algorithm and give it more degrees of freedom, we also create larger cells that we call *dbl* cells. We create *dblmx* and as well as *dbl*1*out* variants. Their structure and purpose is the same as their *sngl* counterparts except that they use larger device sizes ($\leq 2\times$ of their *sngl* counterparts). They are sized such that their delays are closer to the delays for regular gates.

All these variants are crucial to our approach and help provide enough flexibility to our algorithm, reducing the leakage of a given circuit while making sure that there is *no delay penalty*. The details of the algorithm are explained in Section IV-D.1.

IV-D.1. The Gate Replacement Algorithm

Before we use the gate replacement algorithm, we first characterize our library of cells (including the variants) using SPICE [32], and generate a file in the GENLIB [33] format from the characterized data. In the GENLIB format, each pin of a gate is associated with an intrinsic delay component as well as a load dependent component for both rise and fall times. Also included in the *genlib* file is the load capacitance of each input pin.

The pseudo-code for our algorithm is shown in Figures IV.2 and IV.3. Our algorithm takes as input a netlist of gates in levelized order. We first perform a static timing analysis on this netlist to find the Arrival Times (ATs) and Required Times (RTs) at all nodes in the circuit. We use the cell characterization data (which accounts for the load dependency of both the rising and falling delays of the gates) for our static timing analysis . We assume that for gates driven by primary inputs, the primary input can be split to set the desired logic value at the inputs of these gates. Once the logic values of the inputs to the 0^{th} level of gates (the gates with only primary inputs as the inputs) has been fixed, we propagate these values forward to the next level. Next, we pick a gate G from the first level. Lets say the output of the gate is a signal g. We then search through each of the gates h in the fanout of G and find the value of g that gives the minimum possible leakage for h. From this we get the logic value required of g for each h. For example, if one of these fanout gates is a 2 input gate H and assuming that one of its inputs is set to 1 due to another gate J, we would pick the minimum leakage from the following set of input vectors (11, 10). Thus we get the value of g required to get this 2 input gate H in its minimum possible leakage state. Note that when we first visit any gate, we assume all possible input vectors are possible at each gate (i.e. we would consider all vectors 00, 01, 10 and 11 to get the minimum possible leakage vector). This step of finding the best value of g is done for all fanouts of G. If we need to set the value of g to 0 for some fanouts and to 1 in others (which

Algorithm replaceGateForMinLkg (levelized_netlist, genlib_data, allowed_slack)

find AT at all nodes

find RT at all nodes

set all gates at first level to minimum leakage state

for ($i = 1; i <= maxLevel\_of\_Ckt; i++$) **do**

  for ($j = 1; j <= num\_of\_gates\_at\_Level(i); j++$) **do**

    $G = G(j)$; pick a gate G from the gates at level i

    g = output signal of G

    find suggestedVal of g for all fanout(G)

    if all suggestedVal = 0 and logic value of g = 1 **then**

      $G_{new} = sngl1out_0$ variant of G

      CheckIfReplaceable(G, G_{new})

    else if all suggestedVal = 1 and logic value of g = 0 **then**

      $G_{new} = sngl1out_1$ variant of G

      CheckIfReplaceable(G, G_{new})

    else

      $G_{new} = snglmx_0$ variant of G

      CheckIfReplaceable(G, G_{new})

    end if

  end for

end for

Fig. IV.2. Algorithm to perform gate replacement

Algorithm CheckIfReplaceable (G,G_{new})

Check if G can be replaced by a *sngl* variant

if G can be replaced by *sngl* variant of G reduction in leakage and satisfying timing **then**

replace G with the *sngl* variant

else if G can be replaced by *dbl* variant of G with reduction in leakage and satisfying timing **then**

replace G with the *dbl* variant

end if

Fig. IV.3. Algorithm to check to see if a gate is replaceable

would happen, for example, in situations where the signal g is an input to a NAND gate and a NOR gate), then we check if we can replace the gate G with its *snglmx* variant. We first estimate the leakage savings (if any) of doing this replacement. The presence of the MUX and the weak pull-up/ pull-down used in the *snglmx* variant is a source of additional leakage. However, this increase could be outweighed by the leakage savings at the gates in the fanout of G. We estimate the difference and if there are savings, we then test if replacing G with a *snglmx* variant causes timing violations. If there are timing violations, we attempt to use a *dblmx* variant. Again we first check for leakage savings and if there are savings in leakage, we then check for timing violations. When checking for timing violations due to replacing G with a gate G', we first propagate new RTs at the gate G to its fanins. Also, note that replacing G implies changes in the capacitance seen by the gates in the fanin of G. We then recalculate the AT of the gates in the fanin of G. If the new AT is greater than the new RT, then we do not replace G with G'. If there is no timing violation (there is enough slack) and there are savings in leakage, then replace the gate G with its *dblmx* variant. We follow a similar procedure if all the fanouts of G require the same value at g for minimum leakage. If this value required is the same as the value at g due to fixing

the logic values at the inputs of G, then we don't need to replace the gate. If however, these value differ, then we attempt to first replace the gate with its $sngl1out$ variant. If such a replacement does not reduce leakage current, then we don't replace the gate G and move on to the next gate in the netlist. If such a replacement does not work due to timing slack violations, we then check if a $dbl1out$ variant of G would help without sacrificing power or timing. In this way we traverse the netlist in levelization order from primary inputs to primary outputs and replace gates as we move along, reducing leakage *while guaranteeing that there are no timing slack violations*. In some technologies, gate leakage can contribute to the total leakage. This would only change the leakage table lookup values and not affect the implementation of the algorithm.

IV-E. Experimental Results

We performed extensive experiments to validate our method and compare its results to the minimum circuit leakage values. We simulated the circuits for 10000 random vectors to find the minimum leakage (as suggested in [15]). Simulating 10000 random vectors gives us over 99% confidence that less than 0.5% of the vector population has a leakage lower than the minimum leakage found through this random search. We assumed a library with the following basic cells: INV1X, INV2X, NAND2, NAND3, NAND4, NOR2, NOR3. The circuits for our simulations are from the ISCAS85 and MCNC91 benchmark suites. We first performed a technology independent synthesis on these circuits in SIS [33] using *script.rugged* before mapping it with our library.

In Table IV.2, Column 2 and Column 3 show the minimum leakage current in nA for the original circuit and for the circuit modified by our algorithm, respectively. The % decrease in leakage current is shown in Column 4. The decrease in leakage current is 29.18% on average. *Note that this is the leakage decrease compared to the leakage*

Table IV.2. Leakage, delay improvements and runtimes for our approach

Ckt.	Original Min Lkg (nA)	New Min Lkg (nA)	% Lkg Decr	Original Delay	New Delay	% Delay Incr	Runtime (s)
alu2	1251.72	1022.44	-18.32	1460.70	1422.16	-2.64	5.53
alu4	2598.14	2094.99	-19.37	1755.99	1753.09	-0.17	21.16
apex6	2743.08	1753.82	-36.06	739.94	739.93	-0.00	20.03
apex7	812.72	592.88	-27.05	704.11	704.11	0.00	2.89
C1355	2003.61	1697.87	-15.26	930.41	930.23	-0.02	7.8
C432	584.46	449.93	-23.02	1110.89	1110.89	0.00	1.03
C880	1375.73	977.07	-28.98	1803.93	1718.75	-4.72	6.12
C1908	1909.95	1548.12	-18.94	1489.95	1488.61	-0.09	10.1
C3540	4079.92	3126.00	-23.38	1870.95	1870.63	-0.02	51.89
C6288	13020.10	12011.39	-7.75	5651.08	5637.02	-0.25	695.85
dalu	3293.89	2378.24	-27.80	1506.29	1504.32	-0.13	42.75
des	15218.02	12013.16	-21.06	3021.52	2470.33	-18.24	655.38
i10	8738.32	6318.98	-27.69	2549.68	2499.43	-1.97	238.13
i1	158.38	102.96	-35.00	353.61	353.21	-0.11	0.11
i2	372.66	98.72	-73.51	392.98	392.98	0.00	0.51
i3	323.05	60.13	-81.39	182.46	182.46	0.00	0.98
i6	1907.06	1650.16	-13.47	1080.10	1080.10	0.00	5.5
i7	2499.20	1973.08	-21.05	1088.31	1088.31	0.00	10.38
i8	3805.49	2321.63	-38.99	1591.76	1297.01	-18.52	38.62
i9	2552.20	1440.26	-43.57	1651.78	1618.21	-2.03	15.87
t481	2915.54	2409.63	-17.35	901.69	838.36	-7.02	28.21
too_large	1034.72	796.34	-23.04	680.24	677.89	-0.35	4.09
Avg			-29.18			-2.56	84.68

obtained by applying input vector control alone.

The critical delays (in ps) for the original and the modified circuit are shown in Columns 5 and 6 respectively. Column 7 gives the % decrease in critical delays of the modified circuit. We conjecture that one of the reasons for the delay decreasing is due to the fact that when the algorithm can't choose a *sngl* variant due to timing issues, it chooses a *dbl* variant and this can cause a decrease in the delay. Also, as mentioned in Section IV-D, while the delay of one type of transition gets worse in the *sngl1out* variants, the delay of the opposite transition is sped up slightly. The last Column of Table IV.2 reports the runtimes of the algorithm. The algorithm is currently implemented in PERL and was run on an Intel Pentium 4 with 2GB of RAM, running Linux Fedora Core 3. The runtimes are expected to improve substantially when the algorithm is implemented in a compiled language such as C/C++.

Our algorithm assumes that there are MUXes at the primary inputs. They help ensure that all 0^{th} level gates can be set independently into their low leakage state. For a fair comparison, we give the same flexibility (ability for the inputs of each of the 0^{th} level gates to be set independently) when finding the minimum leakage vector for the original circuit.

Table IV.3. Area (active area) cost of using our approach

Ckt.	Original Area(μ^2)	Total New Area(μ^2)	Total New Area Ovh (%)	Sleep Transistor Area(μ^2)	New Area excluding sleep cut-off transistors (μ^2)	Area overhead excluding sleep cut-off transistors (%)
alu2	78.52	96.20	22.52	14.08	82.12	4.58
alu4	155.42	187.94	20.92	24.87	163.07	4.92
apex6	157.36	197.15	25.29	34.71	162.44	3.23
apex7	49.04	66.32	35.24	15.05	51.27	4.55
C1355	108.20	133.74	23.60	22.34	111.40	2.96
C432	37.92	46.01	21.33	7.29	38.72	2.11
C880	83.94	107.56	28.14	20.52	87.04	3.69
C1908	104.21	134.74	29.30	26.95	107.79	3.44
C3540	246.42	305.13	23.83	48.84	256.29	4.01
C6288	672.99	970.35	44.18	260.06	710.29	5.54
dalu	211.55	259.04	22.45	38.50	220.54	4.25
des	812.09	1054.80	29.89	209.27	845.53	4.12
i10	490.08	621.40	26.80	109.84	511.56	4.38
i1	11.90	13.99	17.56	1.85	12.14	2.02
i2	50.84	53.99	6.20	2.81	51.18	0.67
i3	32.28	40.36	25.03	5.00	35.36	9.54
i6	109.22	124.21	13.72	13.49	110.72	1.37
i7	147.63	170.96	15.80	21.11	149.85	1.50
i8	234.59	273.09	16.41	32.37	240.72	2.61
i9	151.56	179.53	18.45	24.13	155.40	2.53
t481	166.08	213.81	28.74	40.15	173.66	4.56
too_large	62.51	80.85	29.34	15.40	65.45	4.70
Avg			23.85			3.69

In Table IV.3, the area penalty associated with using our algorithm is given. Note that this table refers to only the active area. Column 2 of the table shows the area of the original circuit. Column 3 and Column 4 of the table give the total area and the area overhead respectively of the modified circuit including the area of the sleep cut-off transistors used in the $sngl1out$ and the $dbl1out$ type of gates. The active area of these sleep cut-off transistors is reported in Column 5. Column 6 (which is obtained by subtracting Column 5 from Column 3) and Column 7, report the area and area overhead respectively of the modified circuit *excluding* the sleep-cut-off transistors. On average, the total active area overhead

including the sleep cut-off transistors is about 23.6%. However, the active area overhead excluding the sleep cut-off transistors is only about 3.7% which implies that the sleep cut-off transistors caused most of the active area penalty. The size of the sleep transistors can be reduced by sharing them as is done in many MTCMOS based designs. This would not only save area but also reduce leakage. Hence, we consider the active area *excluding* the sleep-cut off transistors (Columns 6 and 7 of Table IV.3) to be a more meaningful measure of the area penalty. Another important point to note is that the area overhead reported is only the active area overhead. The effective area overhead is expected to be much smaller once the circuits are placed and routed.

We also estimated the dynamic power consumption associated with using our approach. Intuitively, the dynamic power overhead is expected to be proportional to the active area overhead excluding the sleep transistors (3.7%). However, some of this active area is devoted to the sleep bypass transistors which contribute only their diffusion capacitance to the total switched capacitance during circuit operation. Based on this we estimated the total switched capacitance overhead which is proportional to the dynamic power consumption overhead. The switched capacitance overhead is shown in Column 8 of Table IV.4. The average switched capacitance overhead is only about 1.5% which is also roughly the dynamic power consumption penalty. Table IV.4 also shows statistics of the type (or variant) of the replacement gates used. We find that the *dblmx* variant of the gates did not get used at all. The *sngl1out* was the variant that was used the most. The next variant used most often was the *snglmx* variant. This variant along with the *dblmx* variant, are the variants that offer the most flexibility in controlling the internal node voltages.

Tables IV.2, IV.3 and IV.4 validate the effectiveness of our methodology. Note, that the modified circuits have a lower leakage with *no delay penalty* (or in some cases a delay improvement) and a very small increase in dynamic power consumption. This is an improvement over previous approaches [39, 40, 41, 42] that obtain similar leakage im-

Table IV.4. Statistics of replacement gates utilized and switched capacitance overhead of using our approach

Ckt.	#sngl1out	#dbl1out	#snglmx	#dblmx	Total # replacements	Total # of gates	Switched cap Ovh.(%)
alu2	91	0	30	0	106	374	2.42
alu4	183	2	66	0	218	713	2.68
apex6	204	0	18	0	213	779	1.06
apex7	94	0	6	0	97	255	1.38
C1355	91	16	0	0	107	582	1.38
C432	40	0	0	0	40	170	0.42
C880	119	0	12	0	125	404	1.31
C1908	150	3	6	0	156	548	1.04
C3540	327	0	58	0	356	1174	1.69
C6288	1649	2	70	0	1686	3578	1.53
dalu	342	0	36	0	360	946	1.53
des	1171	0	170	0	1256	4169	1.64
i10	736	2	112	0	794	2421	1.79
i1	12	0	0	0	12	52	0.40
i2	17	0	0	0	17	171	0.13
i3	4	60	0	0	64	114	6.37
i6	75	0	0	0	75	586	0.27
i7	111	0	0	0	111	719	0.30
i8	266	0	14	0	273	1102	0.75
i9	167	2	4	0	171	735	0.73
t481	237	0	48	0	261	803	2.05
too_large	89	0	20	0	99	304	2.17
Avg	280.68	3.95	30.45	0.00	299.86	940.86	1.50

provements but at the expense of a delay increase. Our technique does not require multiple threshold voltages (which are required in MTCMOS based methodologies) or multiple supply voltages (which are required in VTCMOS based methodologies). Also, our technique does not suffer from the high currents drawn and the spurious transitions that occur when a MTCMOS circuit wakes up from the sleep mode. This is because in our technique, internal nodes do not float (outputs of gates are at full-rail values) when the circuit is put into the sleep state. In MTCMOS circuits, internal nodes float when the power gating sleep transistors are turned off.

We also performed experiments to test if our algorithm could reduce leakage even further if the allowed timing slack was increased. The results are shown in Table IV.5. We notice, that not too many circuits (some exceptions are apex6, C432 and i9) are able to take advantage of the slack available. Our methodology currently only uses input vector control

and circuit modification to allow control of internal node signals. However, if we allow the replacement of a gate with a lower leakage gate (through device sizing) or if we allow the the reduction of the size of the sleep cut-off transistors, then we could take advantage of the allowed slack. These features are not currently implemented since the primary goal was to decrease leakage with no delay penalty.

Table IV.5. Leakage improvement for different allowed slacks

Ckt.	0% slack		10% slack		20% slack	
	Lkg decr(%)	Delay incr(%)	Lkg decr(%)	Delay incr(%)	Lkg decr(%)	Delay incr(%)
alu2	-18.32	-2.64	-18.07	-2.28	-18.07	-2.28
alu4	-19.37	-0.16	-19.49	5.26	-19.49	5.26
apex6	-36.06	-0.00	-36.28	5.83	-36.21	18.34
apex7	-27.05	0.00	-28.39	6.87	-28.39	6.87
C1355	-15.26	-0.02	-24.08	4.73	-24.08	4.73
C432	-23.02	0.00	-33.13	9.22	-35.53	15.14
C880	-28.98	-4.72	-30.25	-6.45	-30.25	-6.45
C1908	-18.94	-0.09	-19.30	2.38	-19.30	2.38
C3540	-23.38	-0.02	-23.22	5.75	-23.22	5.75
C6288	-7.75	-0.25	-7.54	1.53	-7.54	1.53
dalu	-27.80	-0.13	-27.33	3.32	-27.33	3.32
des	-21.06	-18.24	-21.06	-18.24	-21.06	-18.24
i10	-27.69	-1.97	-27.69	-1.68	-27.69	-1.68
i1	-35.00	-0.11	-41.13	5.54	-41.13	5.54
i2	-73.51	0.00	-76.18	3.70	-76.18	3.70
i3	-81.39	0.00	-90.37	5.86	-90.37	5.86
i6	-13.47	0.00	-25.28	-7.91	-25.28	-7.91
i7	-21.05	0.00	-27.28	-5.61	-27.28	-5.61
i8	-38.99	-18.52	-38.91	-18.52	-38.91	-18.52
i9	-43.57	-2.03	-43.93	7.08	-44.00	11.56
t481	-17.35	-7.02	-17.35	-7.02	-17.35	-7.02
too_large	-23.04	-0.34	-24.11	6.04	-24.11	6.04
Avg	-29.18	-2.56	-30.28	0.25	-30.28	1.29

IV-F. Chapter Summary

In this chapter we presented an algorithm that replaces gates in a circuit, in an effort to reduce the standby leakage of the circuit. This replacement does not necessarily reduce the leakage of a gate being replaced, but helps set the gates in the transitive fanout to their low leakage states. The algorithm involves traversing the circuit from the PIs to the POs, replacing gates as required to try and set as many gates as possible to their low leakage

state. We get an average decrease in leakage of about 29% with an active area penalty of about 24%. This leakage decrease is the decrease over the leakage obtained through input vector control alone.

Possible extensions to this work could be using a larger library with complex gates and implementing a 'smarter' algorithm that starts with a solution (given an initial minimum leakage vector) and then replaces gates if required. This could potentially yield much lower leakage currents.

CHAPTER V

OPTIMUM REVERSE BODY BIASING

V-A. Chapter Overview

One of the methods to reduce leakage power is by increasing the threshold voltages (V_T) of the device. This is done either statically, through use of multi-threshold devices or dynamically, through Reverse Body Biasing (RBB).

The sub-threshold leakage (cut-off) current of a transistor decreases with greater applied RBB. Reverse Body Biasing affects V_T through body effect, and sub-threshold leakage has an exponential dependence on V_T, as we have discussed earlier.

However, while the sub-threshold leakage decreases, there are other components to the leakage current that have to be considered as well. Two of these are bulk Band-to-Band-Tunneling (BTBT) and surface BTBT. Bulk BTBT is commonly referred to as simply BTBT while surface BTBT is commonly called Gate Induced Drain Leakage (GIDL) [43, 44]. While GIDL does not play a major role at RBB [43], BTBT increases with applied RBB [43, 45, 46, 47]. This means that there is an optimum RBB voltage at which the *total* leakage power (the sum of the sub-threshold leakage, the gate leakage, BTBT and GIDL) is minimum [43, 45, 46, 47]. In modern processes this optimum point is reached before the upper limit of the RBB (based on the voltage at which the bulk-drain / bulk-source junction breaks down). Also, this optimum point can vary with temperature and process variations. In this chapter we show that it is desirable to operate at the optimal RBB point which minimizes total leakage. We present a scheme that monitors the total leakage current (the sum of the sub-threshold, BTBT and gate leakage) of an IC with a representative leaking device and, using this monitored value, *automatically* finds the optimum RBB value across temperature and process corners, using a self-adjusting circuit. Our approach has a

modest placed-and-routed area utilization, and a low power consumption. In section V-B we discuss the motivation behind our work. Section V-C discusses previous approaches to dynamically adjust body-bias. Section V-D describes our approach to dynamically self-adjust the RBB of PMOS and NMOS devices in order to obtain a minimum total leakage, along with experimental results that support the utility of our scheme.

V-B. Goal and Background

In this work we are concerned with minimizing the total leakage current (the sum of the sub-threshold, BTBT and gate leakage) through a non-conducting (turned-off) device in a static CMOS design. In the case of an NMOS device this would mean we are concerned with minimizing the leakage (over possible RBB values) through an NMOS device when its drain terminal is at VDD, its source and gate terminals are at GND and its bulk terminal (p-well) is at a certain RBB value. In such a scenario, the leakage current measured at the drain of the device is mainly due to three sources – (i) the sub-threshold leakage from the drain to the source of the device, (ii) the gate leakage current from the drain to the gate and (iii) the drain-bulk junction current. The drain-bulk leakage current has three main components – bulk BTBT (or simply BTBT), surface BTBT (or GIDL) and the classical reverse biased PN junction current [48, 2, 43] (see Figure I.2 in Chapter I). The bulk BTBT current is also often referred to as Gate Edge Drain Leakage (GEDL). This current is due to the tunneling of electrons from the valence band of the p-region (from the bulk) to the conduction band of the n-region (to the drain). This tunneling happens due to a high electric field across the bulk-drain junction (which can happen when a Reverse Body Bias (RBB) is applied). Gate Induced Drain Leakage current (GIDL) occurs when the gate bias is negative relative to the drain [2, 49]. At negative gate bias, the overlap region of the gate and drain gets depleted of carriers. Minority carriers (generated by BTBT and other tunneling mechanisms) arrive at

the surface to attempt to form an inversion layer in the channel and are immediately swept laterally to the substrate. Due to the field across the gate and bulk junction, these carriers then flow into the bulk node. This current is the GIDL current.

The two BTBT currents dominate the reverse biased PN junction current. While the sub-threshold leakage decreases with increased RBB (due to the increase in V_T of the device), bulk BTBT current increases with RBB. The BTBT current density equation [50] is given below

$$J_{BTBT} = A\frac{EV_{app}}{\sqrt{E_g}}e^{-B(\frac{E_g^{3/2}}{E})} \tag{5.1}$$

$$A = \frac{\sqrt{2m^*}q^3}{4\Pi^3\hbar^2} \tag{5.2}$$

$$B = \frac{4\sqrt{2m^*}}{3q\hbar} \tag{5.3}$$

In these equations, m^* is the effective mass of an electron, E_g is the energy band-gap, V_{app} is the applied reverse bias, E is the electric field at the junction, q is the electron charge, and $\hbar = 1/(2\Pi)$ times Planck's constant.

Assuming a step function, the electric field at the junction is

$$E = \sqrt{\frac{2qN_aN_d(V_{app}+V_{bi})}{\varepsilon_{Si}(N_a+N_d)}} \tag{5.4}$$

where N_a and N_d are the doping in the P and N devices, ε_{Si} is the permittivity of silicon and V_{bi} is the built-in voltage across the junction. Hence for a step junction, J_{BTBT} is approximately proportional to $V_{app}^{3/2}$. However, the exact dependence of E on V_{app} varies with the doping profile of the substrate [45].

The drain-gate leakage current does not change appreciably with applied RBB [45]. Also, at RBB, bulk BTBT dominates GIDL [43]. Hence it is mainly the sub-threshold and the BTBT component of the leakage currents that change with applied RBB. Also, since these two components behave differently with respect to RBB, there exists an optimal

RBB value [46, 2, 43, 45] which minimizes leakage. We performed experiments on a test-chip manufactured using the TSMC 0.13μm triple well process to find the RBB value that minimizes total leakage. The test chip had one large PMOS (W_{eff} = 676mm, L_{eff} = 0.13μm) and one large NMOS (W_{eff} = 504mm, L_{eff} = 0.13μm) device. The devices on the test chip were made large so that their different leakage current components would be easy to measure. The drain, source, gate and bulk contacts were all brought out as pins, enabling us to measure the currents at each of these contacts. When a device is turned-off, the current measured at the source represents the sub-threshold leakage current from the drain to the source (I_{ds}), the current measured at the gate represents the gate leakage from the drain to the gate (I_{dg}) and current measured at the bulk contact represents the drain/source to bulk current (I_{db},I_{sb}). Since the drain is at VDD, most of the bulk current is from the drain (i.e. I_{db} dominates I_{sb}). The current measured at the drain of the device (I_{leak}) was found to be approximately the sum of the currents measured at the gate, source and bulk terminals confirming that I_{sb} is very small in practice.

Figure V.1 shows measurements taken from our manufactured test chip for a non-conducting NMOS device at a temperature of 25° C with the RBB being swept from 0.7v to 1.1v below the source terminal. The VDD used was 1.2v. In this case the optimal RBB value is 1.0v.

Table V.1. Leakage penalty due to temperature variation

Temp (°C)	Lkg penalty
-40	23.38%
0	6.99%
25	0%
70	35.29%
125	163.55%

The optimum RBB value can shift with temperature and process variations. Table V.1 shows the penalty due to temperature variations (in terms of percentage of leakage power

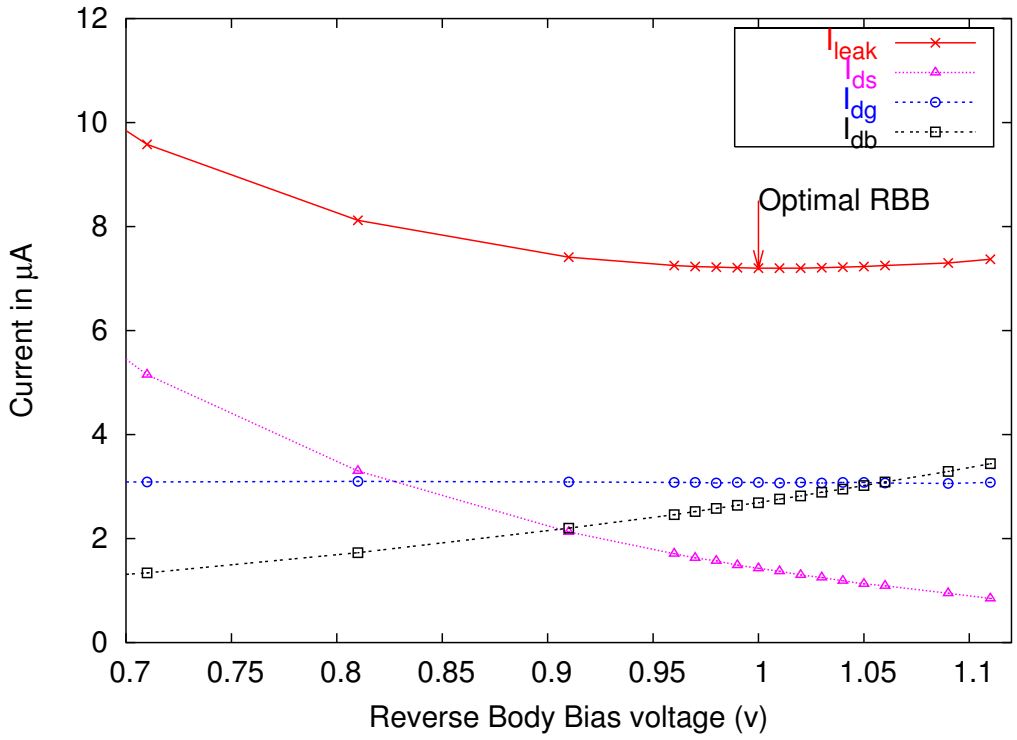


Fig. V.1. Leakage current components for a large NMOS device at 25°C

increase from optimum) for the large NMOS device, while Table V.2 reports the penalty due to process variations, assuming that the RBB is fixed to the optimum value (1.015V) for one particular temperature and process corner (25°C and nominal corner in this case).

Tables V.1 and V.2 prove that fixing the RBB at a particular value may not be a good idea if we are interested in reducing leakage over *all* temperature and process variations. We hence need a scheme by which we can monitor the leakage current of a chip and automatically self-adjust the RBB value of the PMOS and NMOS devices, to keep the leakage power as low as possible. The problem of monitoring the optimum point is compounded by the fact that the total leakage current can vary by as much as 3 orders of magnitude over temperature and RBB variations. The leakage monitor must therefore be able to find the optimum RBB point over this wide range of currents.

Table V.2. Leakage penalty due to process (V_T, l_{eff}) variation

V_T	l_{eff}	Lkg penalty
nominal	nominal+10nm	16.15%
nominal	nominal-10nm	4.02%
nominal	nominal	0%
nominal-8%	nominal-10nm	10.73%
nominal+8%	nominal+10nm	58.3%
nominal+8%	nominal	20.77%

V-C. Related Previous Work

In [45], a simple circuit is presented that helps find the optimal RBB value. The accuracy of this circuit is dependent on the assumption that gate leakage can be neglected (or is very small) and that sub-threshold leakage is negligible when compared to the BTBT current in a stack of 2 non-conducting devices. Under these assumptions, the authors claim that the optimal RBB value occurs at the point where the leakage current through two stacked non-conducting devices is primarily BTBT current, and is equal to half the leakage through a single non-conducting device. However, experiments with our test chip show that these assumptions are significantly inaccurate.

Figure V.2 shows a plot of *half the leakage current* through a single non-conducting NMOS device on our test-chip (labeled as 'Id single div 2') and the leakage current through a stack of two non-conducting NMOS devices (labeled as 'Id stack'). The currents were measured at a temperature of 25° C. The arrow labeled 'A' shows the optimal RBB value as would be suggested by the circuit in [45] while the arrow labeled 'B' shows the actual optimal RBB value for a single non-conducting NMOS device at 25°C. We found that if the RBB value marked by A was used as the "optimal" RBB instead of the RBB value pointed by B, the leakage current for a single non-conducting NMOS device (at 25°C) would be 70% higher than optimum.

In [11] and [51] the authors suggest sensing the voltage dropped by a leaking device

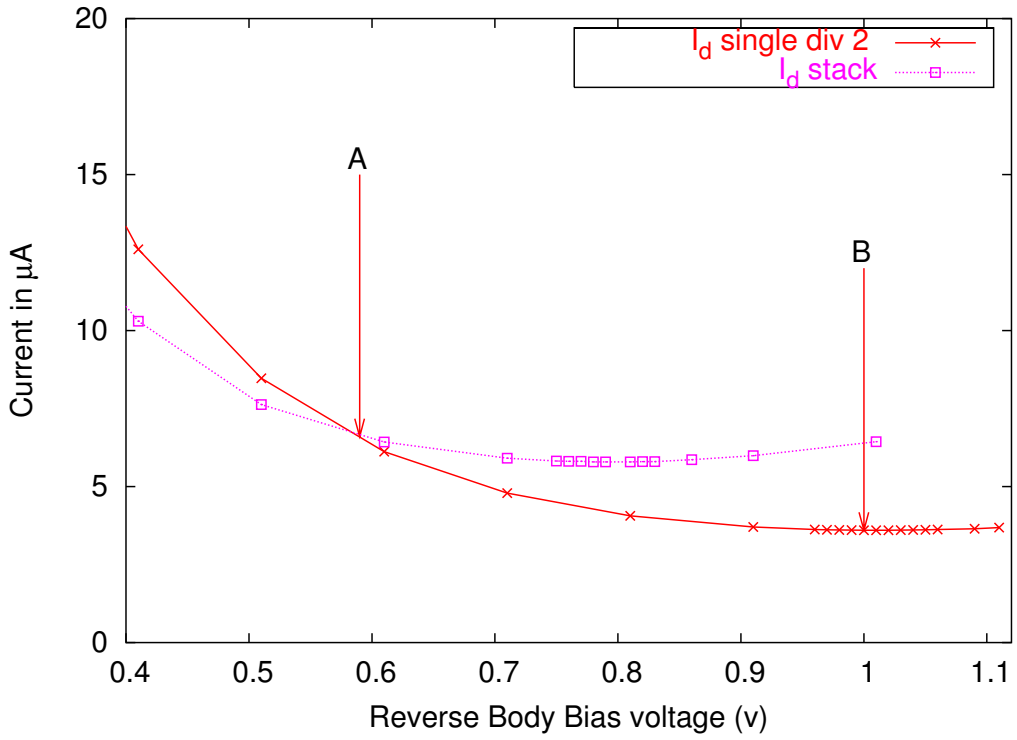


Fig. V.2. Leakage current for stacked and single devices

towards the goal of adjusting the body bias and thus controlling the leakage. To amplify the leakage current, the gate bias is set to a value such that the leaking device is still cut-off but has a high enough leakage current to drop a significant voltage. This voltage is sensed and if it crosses a certain threshold, RBB is applied. The authors of [52] suggest a similar mechanism as a way of stabilizing sub-threshold CMOS logic. However, [11, 51, 52] do not target the problem of finding the *optimum* RBB value.

V-D. Leakage Monitoring / Self-Adjusting Scheme

Our leakage monitoring scheme is based on measuring the time taken for the leakage current to discharge (for monitoring the leakage of a leaking NMOS device) a capacitive load. For a leaking PMOS device, the time taken for charging-up the load is considered. A higher

leakage would be indicated by a shorter time to discharge the load while a longer time to discharge the load would indicate a lower leakage. To monitor the leakage current of an NMOS device, the capacitively loaded node is initially pre-charged to a logic-high value. The leakage current is estimated by measuring the time taken to discharge this node. Similarly, for a leaking PMOS device, the capacitively loaded node is initially pre-discharged and the leakage current is estimated based on the time taken to charge this node to a logic-high value.

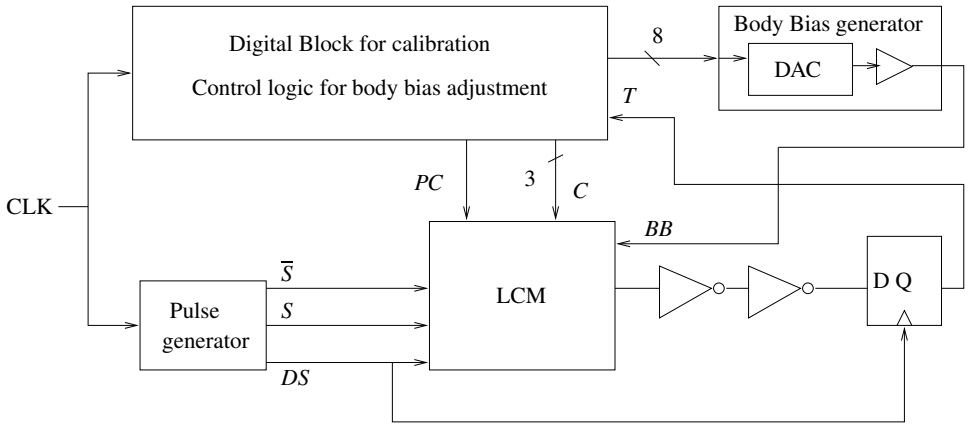


Fig. V.3. LCM scheme block diagram(for NMOS)

The leakage monitoring scheme is conceptually illustrated in Figure V.3 (for NMOS bulk control). A similar structure is used to control the PMOS bulk node. The 3 main blocks of the leakage monitoring scheme are: (i) a leakage current monitoring (LCM) block that contains a representative leaking device, (ii) a digital block to interface with the LCM and control the body bias voltage and (iii) a programmable body bias voltage generator to translate the body bias control value from the digital block into a body bias voltage value. In this chapter we deal with the leakage monitoring block and the digital control block. Details of the bias generator are omitted, ant it is assumed that this function

is performed by an off-the-shelf Digital to Analog converter (DAC) IC.

V-D.1. Leakage Current Monitoring Block (LCM)

In this section the design and operation of the LCM block will be discussed. We use the LCM for NMOS devices as an example. Our objective is to track the variation of total leakage current through a circuit with applied RBB. However, placing a current monitoring device in series with the IC supply and circuit power rails of the logic devices is not an option since the addition of such a device would increase the delay of the circuit. Hence we choose a representative device to model the leakage of the entire circuit. The optimal RBB value is smaller for stacked devices when compared to single (unstacked) devices. This is because sub-threshold leakage is lower for stacked devices and hence BTBT dominates at a lower RBB value. However, it is infeasible to have separate substrates for stacked and non-stacked devices. In our scheme we chose a non-stacked device as the representative leaking transistor based on the intuition that for most ICs the dominant source of leakage is from unstacked devices. However, if we were to design a leakage monitor to track the leakage of an IC (with stacked devices being the dominant source of leakage), the leakage monitor would have to use stacked devices as the representative leaking transistors.

The leakage current variation of NMOS and PMOS devices is monitored separately. Figure V.4 shows the circuit that implements the leakage current monitoring block for NMOS devices. In Figure V.4, device M_L is the representative leaking transistor. Transistor M_{pchg} is the device that precharges the node N_{chk}. M_L and M_{pchg} are sized relatively so that the leakage of M_L dominates the leakage of M_{pchg}. The leakage monitoring scheme is based on the idea that the time taken for the leaking transistor M_L to discharge the node N_{chk} would be proportional to the leakage current through M_L and hence the leakage current through the entire circuit.

In Figure V.4, the capacitor bank and the device M_{gpd} allow the LCM to work over a

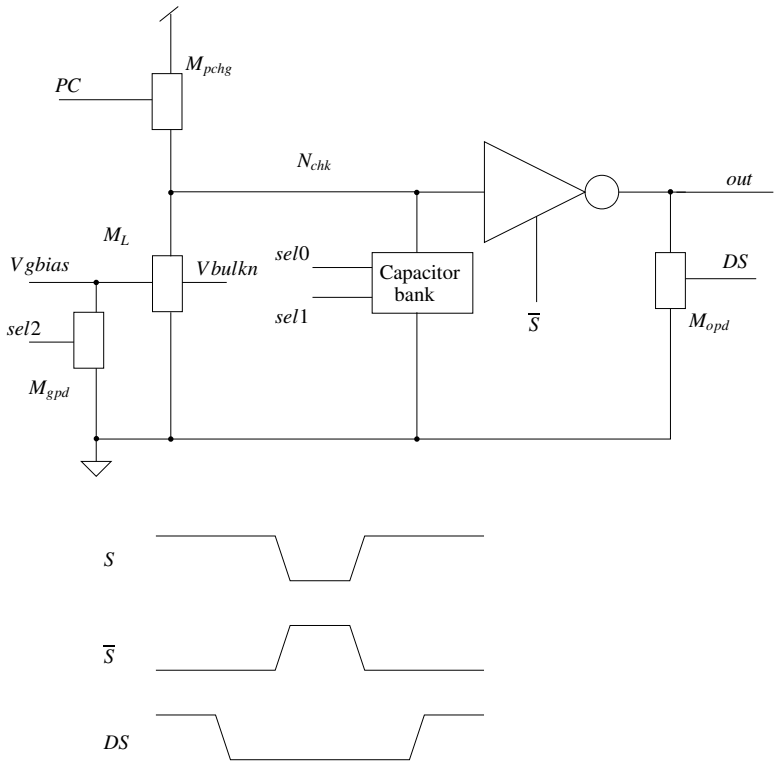


Fig. V.4. LCM for NMOS devices

wide range of leakage currents. If the leakage current is too low, it needs to be magnified for the LCM to work effectively. This is done by first disconnecting the capacitor bank from N_{chk} (to speed up the rate of discharge of the node N_{chk}). Further magnification of the leakage current is achieved by turning off M_{gpd} and hence increasing the gate bias of M_L (in a similar manner as in [11, 51]) to a value of about 0.1V above GND (such that M_L is still in the sub-threshold/cut-off mode).

The circuit that generates this low gate bias voltage is designed such that its output voltage decreases with an increase in temperature. Without this feature, the current in M_L increases too rapidly with increasing temperature when M_{gpd} is off.

The LCM works by 'sampling' (turning on the tri-stateable inverter at the output of the LCM) the node N_{chk} at regular intervals. During this sampling, the output pulldown device, M_{opd} is turned off. Note that the sampling period is short, which keeps the power consumption of the LCM low. If the node N_{chk} has fallen low enough, the output of the LCM goes high and this output is buffered and then latched in a D flip-flop. The DFF output (shown as T in Figure V.3) triggers the digital block. The purpose of this trigger signal will be explained in the following sub-section.

The LCM for PMOS devices is implemented in a manner similar to that of the LCM for NMOS devices.

V-D.2. Digital Control Block

The Digital Control Block contains an 8-bit counter that counts up till either the end of the count is reached or till it receives a trigger signal from the DFF at the output of the LCM. When a trigger signal is received, the value of the 8-bit counter is stored. This counter value is proportional to the time taken for the transistor M_L to discharge the node N_{chk} and is hence a measure of the leakage current of M_L. Next, the node N_{chk} is precharged (signal PC goes low) and held in this precharged state till a new body-bias is set. The applied RBB value is increased till the point at which the new counter value is smaller than the previous counter value (the point at which the leakage current starts increasing with applied RBB). If the end of the count is reached before a trigger signal is received, this implies that the total leakage is too low. In such a situation, control signals from the digital block are applied to the LCM to magnify the leakage current. The digital block sends appropriate signals (shown as C in Figure V.3 and $sel0$, $sel1$, $sel2$ in Figure V.4) that control the capacitor bank and M_{gpd} in the LCM to achieve this magnification, as described in Section V-D.1.

In summary, our leakage monitoring scheme works by essentially converting the problem of sensing the total leakage current into one of measuring the time taken for a represen-

tative leaking transistor to discharge a purely capacitive load. The time taken is measured using a counter and the applied RBB is increased in linear steps till the time measured by the counter for a particular body-bias value, is shorter than the time measured by the counter for a previous body-bias value used. The LCM is designed for correct operation over a wide range of leakage currents.

The accuracy of the scheme can be improved by increasing the frequency of the clock and hence increasing the frequency of sampling of the node N_{chk}. We utilize a clock with a period of 2ns. Simulations showed the proposed scheme has a very small power consumption of 11.4μA. Of this, the LCM block consumes about 4μA, while the digital control block consumes about 6μA. Note that simulations were done at 1.2V at 125°C (to model the worst-case power consumption) for a TSMC 0.13μm process. The digital block was synthesized using a 0.13μm process standard-cell library.

Table V.3. Size of the standard-cell implementations of the LCMs and pulse generator

Cell	Width(μm)	Height(μm)	Area(μm^2)
LCM NMOS	77.87	3.285	255.7
LCM PMOS	86.41	3.285	283.86
Pulse generator	38.22	3.285	125.55
Total	-	-	665.11

We also created layout macro-cells for the pulse generator (that generates the S and DS signals for the LCM block), the LCM block for NMOS leakage monitoring and the LCM block for PMOS leakage monitoring. The LCM blocks include the circuitry required to generate the low $Vgbias$ voltage. Table V.3 shows the placed-and-routed size of each cell in the layout.

V-E. Chapter Summary

In this chapter, we have described an automatic, self-adjusting mechanism to find the optimal RBB value to minimize total leakage. Our method consists of a leakage current monitor, and a digital block that senses the discharging (charging in the case of a PMOS transistor) of a representative NMOS device in the design. Based on the speed of discharge, which is faster for leakier devices, an appropriate RBB value is applied. Our technique is able to find the optimal RBB point, and incurs very reasonable placed-and-routed area and power penalties in its operation.

CHAPTER VI

EXPLOITING LEAKAGE - SUB-THRESHOLD CIRCUIT DESIGN

VI-A. Chapter Overview

In the first study in this dissertation we discussed the problems faced due to leakage and proposed techniques to minimize leakage. In the next few chapters of this dissertation, we propose techniques to exploit leakage instead of minimizing it. We do this through the use of *sub-threshold circuit design*.

Due to their extreme low power consumption, sub-threshold design approaches are appealing for a widening class of applications which demand low power consumption and can tolerate larger circuit delays. In this chapter, we present the opportunity that sub-threshold circuit design presents.

VI-B. Introduction

The ever-increasing popularity of battery-powered and portable electronics underscores the importance of power consumption as a significant issue in VLSI design. The utility of wearable / portable computing and communication devices hinges on the ability to design low power VLSI circuits efficiently. Further, sensor networks [53, 54, 55] have recently been shown to be a powerful means to gather, process and communicate data in a variety of operating environments. The distributed nature of these networks, along with the need for each sensor to be maximally maintenance-free (ideally sustained by power from ambient light) further underscores the importance of low-power electronics.

In applications such as sensor networks and wearable electronics devices, the speed of operation is not a paramount design consideration. Rather, power reduction (which translates into longer battery life, or reduced system weight resulting from the need for

smaller battery packs) is a major design consideration. A practical approach to designing VLSI ICs with extreme low power consumption would be very desirable for this large and growing class of practical applications. In this dissertation such an approach, using sub-threshold circuits, is described.

As the minimum feature size of processes continues to shrink with each successive process generation (along with the value of supply voltage and therefore V_T), leakage currents increase exponentially. On the one hand this would suggest the use of larger V_T values, but this in turn leads to slower circuits since the device (operating in linear or saturation region) has a slower turn-on when V_T is increased. Choosing a lower V_T results in lower delays but increased leakage power dissipation. Leakage power already comprises about 50% of the total power dissipation of modern designs [4, 56], so this option is not desirable either. Sub-threshold (leakage or cut-off) [7, 6] currents are hence seen as a necessary evil in traditional VLSI design methodologies. In this dissertation, we explore techniques that turn this problem with leakage currents into an opportunity through the use of sub-threshold circuits.

Sub-threshold circuits exclusively utilize sub-threshold (leakage) currents to implement designs. This is achieved by actually setting the circuit power supply VDD to a value less than or equal to V_T. This choice results in dramatically smaller conduction currents and power at the expense of larger circuit delays. In applications such as sensor networks and wearable electronics devices, the speed of operation is not a paramount design consideration. Rather, power reduction (which translates into longer battery life, or reduced system weight resulting from the need for smaller battery packs) is a major design consideration. A practical approach to designing VLSI ICs with extreme low power consumption would be very desirable for this large and growing class of practical applications.

The advantages of a circuit design approach that utilizes sub-threshold conduction are:

- Power is significantly (100-500×) lower
- Circuits get *faster* at higher temperature [57].
- Device transconductance is an exponential function of V_{gs}, resulting in a high ratio of *on* to *off* current in a device stack. As a consequence, circuit noise margins are high.
- Delay gets worse by 10-25×, but the PDP (Power-Delay Product) improves by 10-20×. We also show (in Chapter VIII), that we can obtain an improvement in the Energy-Delay product as well, by operating the circuit in the near-subthreshold region.

The disadvantages of a sub-threshold design methodology are:

- I_{ds} is small, resulting in large delays.
- I_{ds} exhibits an exponential dependence on temperature, requiring circuitry to compensate for this effect.
- I_{ds} is highly dependent on process variations. For example, small changes in V_T result in large changes in I_{ds} due to the exponential dependence of I_{ds} on V_T. We therefore require circuitry to compensate for this effect as well.
- Design methodologies used today to design sub-threshold logic circuits are ad-hoc. A systematic EDA framework for the design of complex digital systems using sub-threshold logic has not been developed.

Applications such as digital wrist-watches and calculators have utilized extreme low power circuitry based on sub-threshold conduction. However, these applications are analog in nature, or implement very simple digital circuits. The design methodologies used are ad-hoc. A systematic EDA framework for the design of complex digital systems using

sub-threshold circuits has not been developed. Our work attempts to do this, and bring sub-threshold digital design into the mainstream of VLSI technology. Any practical sub-threshold methodology must address the problems of i) the variation of sub-threshold circuit delay with temperature and ii) process variations and iii) supply voltage variations. We address these issues in this dissertation.

VI-B.1. The Opportunity

We performed SPICE [32] experiments to compare the delay of a circuit implemented using sub-threshold CMOS logic versus traditional CMOS logic. Our goal was to compare the delay and power values of both schemes, for a given Deep Sub-micron (DSM) process technology.

The device technologies we used were the Berkeley Predictive Technology Model [27] 0.1μm and 0.07μm processes. For these processes, V_{T_N} and V_{T_P} are respectively 0.261V and -0.303V (for the 0.1μm process) and 0.21V and -0.22V (for the 0.07μm process).

Our comparison of traditional versus sub-threshold circuit delays is shown in Table VI.1. For each process, we constructed a 21-stage ring oscillator circuit using minimum-sized inverters. From this circuit, we computed the delay, power and power-delay product for both design styles. Simulations were performed for a junction temperature of 120°C. Observe that for both the *bsim*70 and *bsim*100 processes, impressive power reductions are obtained, and the power-delay product is about 20× improved, over the traditional design style. The delay penalty can be further reduced by applying a slightly positive body bias. When the body is biased to VDD (which is set at V_T in these simulations), the delay can be brought down by a factor of two, while the power-delay product still remains around 10× better. At this operating point, we still achieve upwards of 100× power reductions.

If V_T can be reduced further, the delay improves as indicated by the sub-threshold

Table VI.1. Comparison of traditional and sub-threshold circuits

	Traditional Ckt			Sub-threshold Ckt ($V_b = 0V$)			Sub-threshold Ckt($V_b = VDD$)		
Process	Dly (ps)	Pwr (W)	P-D-P (J)	Delay ↑	Power ↓	P-D-P ↓	Delay ↑	Power ↓	P-D-P ↓
bsim70	14.157	4.08e-05	5.82e-07	17.01×	308.82×	18.50×	9.93 ×	141.10×	14.43×
bsim100	17.118	6.39e-05	1.08e-06	24.60×	497.54×	20.08 ×	12.00 ×	100.96×	8.20×

current equation below.

$$I_{ds}^{sub} = \frac{W}{L} I_{D0} e^{(\frac{V_{gs}-V_T-V_{off}}{nv_t})} [1 - e^{-\frac{V_{ds}}{v_t}}] \tag{6.1}$$

The adjustment of V_T is easily performed during IC fabrication. We conducted experiments (for the *bsim*100 and *bsim*70 processes) to determine the reduction in delay when V_T is reduced. In these experiments, we used the same absolute value of V_T for both PMOS and NMOS devices, and operated the circuit with VDD = V_T. The results are reported in Table VI.2.

We note that for the *bsim*100 process, reducing V_T to 0.17V results in a 29% delay improvement of our sub-threshold ring oscillator (at this point it is about 17.5× the delay of the traditional ring oscillator), while the power consumption remains 323× lower than that of a traditional ring oscillator (the power is about 500× lower when V_T is 0.28V). Note that the power-delay product, an important figure of merit in circuit design, is a healthy 20× better for the sub-threshold circuit. The V_T reduction can, in practice, be achieved statically or dynamically by appropriately biasing the bulk node of the devices. Further, this V_T reduction can selectively be invoked for devices on the critical computation path, yielding faster designs with extremely low power consumption. Similar numbers are noted for the *bsim*70 process. The delay drops to about 12× the traditional circuit delay at $V_T = 0.13V$, with a 100× power improvement and a 8× improved power-delay product.

Figure VI.1 describes the tradeoffs in the choice of VDD for our methodology. We show the sub-threshold current as a function of V_{gs}, for varying V_{ds} values in 5 steps from

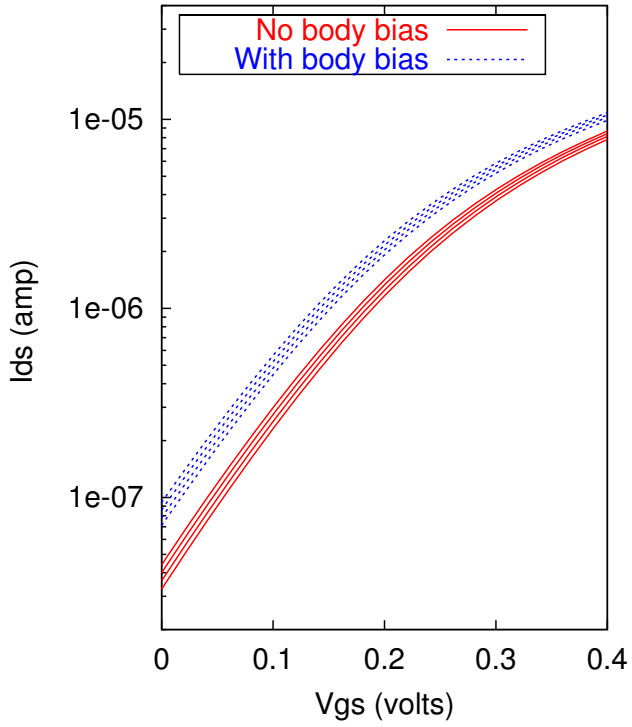


Fig. VI.1. Plot of I_{ds} versus V_{gs} (*bsim70* process)

0 to VDD. We show these currents with and without body bias. Note that for a given V_T, reducing VDD reduces the I_{on}/I_{off} ratio, and hence the circuit becomes less noise immune. At 0.16V, this ratio is about 20, regardless of whether body bias is applied. Note that this means that there is no noise penalty in applying body bias. At higher voltages, this ratio improves, but less than exponentially as we move out of sub-threshold region. Operating at a higher VDD certainly gives us larger switching currents, but the downside is that we have to switch circuit nodes over larger voltage excursions, resulting in quadratically increasing power consumption. On the other hand, operating at a lower VDD (having fixed V_T) results in lower circuit speed but much improved power reduction. For example, for the *bsim*70 process, if VDD = 0.16V (the lowest reasonable value of VDD based on noise considerations), we get a roughly 2× delay penalty and 2× power improvement from the results of

Table VI.2. Sub-threshold circuit delay versus V_T for the *bsim*100 and *bsim*70 processes

*bsim*70				*bsim*100			
V_T	Delay ↑	Power↓	P-D-P ↓	V_T	Delay ↑	Power ↓	P-D-P ↓
0.180	16.15×	167.52×	10.41×	0.270	23.32×	479.85×	20.60×
0.170	14.88×	151.99×	10.09×	0.250	22.43×	464.33×	20.16×
0.160	13.78×	137.73×	9.95×	0.230	21.02×	444.23×	20.05×
0.150	13.15×	124.59×	8.86×	0.210	18.69×	400.89×	20.27×
0.140	12.43×	112.73×	9.40×	0.190	18.42×	366.28×	18.98×
0.130	12.32×	101.85×	8.02×	0.170	17.51×	323.26×	17.98×

Table VI.1.

VI-C. Chapter Summary

In this chapter, we introduced the notion of exploiting leakage currents instead of minimizing them, and presented experimental results that explored the opportunities that sub-threshold circuit design offers. However, sub-threshold circuits have their disadvantages and any feasible approach using sub-threshold circuits must address these disadvantages. In the next few chapters, we propose approaches that do that.

CHAPTER VII

ADAPTIVE BODY BIASING TO COMPENSATE FOR PVT VARIATIONS

VII-A. Chapter Overview

Due to their extreme low power consumption, sub-threshold design approaches are appealing for a widening class of applications which demand low power consumption and can tolerate larger circuit delays. However, sub-threshold circuits are extremely sensitive to variations in power supply, temperature and processing. In this chapter, we present a sub-threshold design methodology which automatically self-adjusts for inter and intra-die process, supply voltage and temperature (PVT) variations. This adjustment is achieved by performing bulk voltage adjustments in a closed-loop fashion. The design methodology uses medium sized Programmable Logic Arrays (PLAs) as the circuit implementation structure. The design has a global *beat clock* to which the delay of a spatially localized cluster of PLAs is ”phase locked”. The synchronization is performed in a closed-loop fashion, using a phase detector and a charge pump which drives the bulk nodes of the PLAs in the cluster. We also present an analysis of the loop gain of this closed-loop adaptive body biasing technique. Our results demonstrate that our technique is able to dynamically phase lock the PLA delays to the beat clock, across a wide range of PVT variations, enabling the sub-threshold design methodology to be applicable in practice.

VII-B. Related Previous Work

In [58, 59, 60], the authors discuss sub-threshold logic for ultra-low power circuits. They state that their approach would be useful for applications where speed is of secondary importance. In one of two proposed approaches, they describe circuitry to stabilize the operation of their circuit across process and temperature variations. In these papers, the idea of

using sub-threshold circuits was introduced from a device standpoint, and candidate compensation circuits were proposed. Also, no systematic design methodology was provided to addresses the multiple issues of process, temperature and supply variations within an IC die.

In [61], the authors report a sub-threshold implementation of a multiplier. The methodology utilizes a leakage monitor, and a circuit which compensates the sub-threshold current across process and temperature variations. In contrast, our approach compensates circuit *delay* directly, by phase locking it to a beat clock. In [62], a dynamic substrate biasing technique is described, as a means to make a design insensitive to process variations. The approach is described in a bulk CMOS context in contrast to our sub-threshold approach. Further, the technique of [62] matches the circuit delay to that of the critical paths (which needs to be found up-front). The dynamic biasing is not performed on a per-region basis, making it susceptible to intra-die variations.

VII-C. Preliminaries - PLAs

In this section we describe the structure and operation of the PLAs used in our approach.

VII-C.1. PLA Design

Consider a PLA consisting of n input variables $x_1, x_2, \cdots, x_n$, and m output variables $y_1, y_2, \cdots, y_m$. Let k be the number of rows in the PLA. A *literal* l_i is defined as an input variable or its complement.

Suppose we want to implement a function f represented as a sum of cubes $f = c_1 + c_2 + \cdots + c_k$, where each cube $c_i = l_i^1 \cdot l_i^2 \cdots l_i^{r_i}$. We consider PLAs which are of the *NOR-NOR* form. This means that we actually implement f as

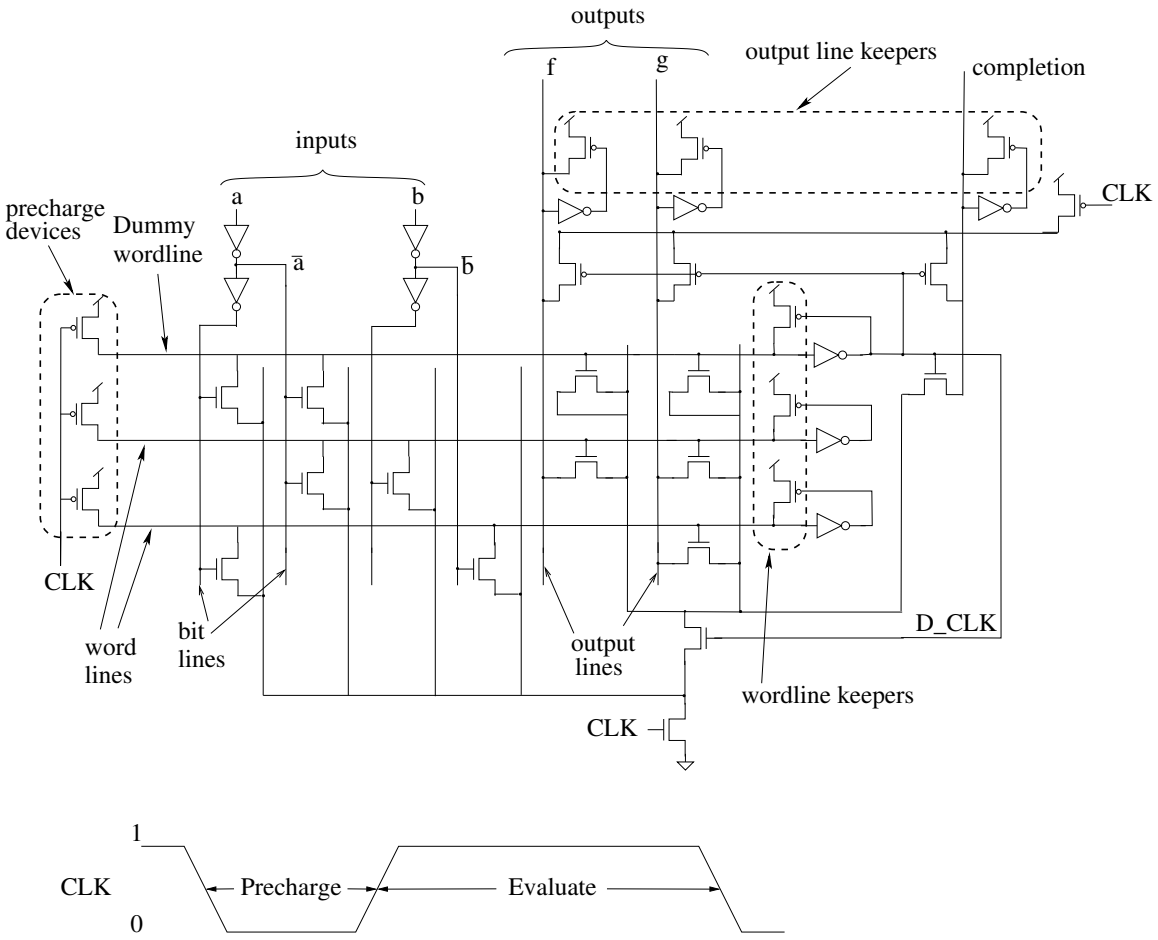


Fig. VII.1. Schematic view of PLA

$$\overline{f} = \overline{\sum_{i=1}^{k}(c_i)} = \overline{\sum_{i=1}^{k}(\overline{\overline{c_i}})} = \overline{\sum_{i=1}^{k}(\overline{\overline{l_i^1} + \overline{l_i^2} + \cdots + \overline{l_i^{r_i}}})} \tag{7.1}$$

The PLA output $\overline{f}$ is a logical NOR of a series of expressions, each corresponding to the NOR of the complement of the literals present in the cubes of f. In the PLA, each such expression is implemented by *word lines*, in what is called the *AND plane*. These word lines run horizontally through the core of the PLA. Literals of the PLA are implemented by vertical-running *bit-lines*. For each input variable, there are two bit-lines, one for each

of its literals. The outputs of the PLA are implemented by *output lines*, which also run vertically. This portion of the PLA is called the *OR plane*.

The PLAs in our design operate in their sub-threshold region of conduction. Figure VII.1 illustrates the schematic of the PLAs used in our design. All the PLAs in our design are of the precharged NOR NOR type, and have a fixed number of inputs (12), outputs (6) and cubes (12).[1] Finally, each output of the PLAs are co-located with a negative edge triggered D flip-flop (DFF), to allow for sequential circuit support. The DFFs are not shown in the Figure VII.1. Since the PLAs evaluate in the high phase of the clock signal, the DFFs are negative edge triggered.

VII-C.2. PLA Operation

The PLAs enter their precharge state when the CLK signal is low. During this time, the horizontal wordlines get precharged. A special wordline (the dummy wordline), which is the maximally loaded wordline also gets precharged. The signal on the dummy wordline is inverted to generate the delayed clock signal D_CLK. When the dummy wordline precharges (after all the other wordlines of the PLA have precharged), the delayed clock D_CLK switches low, cutting off the OR plane from GND. This delayed clock signal is also connected to PMOS pullups at each output line which serve to precharge (pullup) the output lines during the precharge phase. A special output line (which is inverted to produce the signal *completion* shown in Figure VII.1) also gets precharged. The dummy wordline is designed to be the last wordline to switch (by making it maximally loaded among all wordlines). Similarly, the *completion* signal is also the last output signal to switch, since it is maximally loaded as well, in comparison to other outputs. The *completion* signal switching low signals the completion of the precharge operation of the PLA. In the precharged state,

[1] This was found to be a good size from a delay and area point of view for a set of benchmark circuits [63].

all the wordlines and the output lines of the PLA are precharged. Now, when the CLK signal switches high, the PLA enters the *evaluation* phase. In evaluation, if any of the vertical bitlines are high, the wordline that it is connected to, gets pulled low. One of the inputs *and* its complement is connected to the dummy wordline, so that the dummy wordline switches low during *every* evaluate phase and effectively acts as a timing reference for the PLA. By design, the dummy wordline is the last wordline to switch low. When the dummy wordline switches low, it makes the signal D_CLK switch high, as a result of which the GND gating transistor in the OR plane now turns on[2]. The output lines to which, wordlines that have switched low are connected, will switch low. The *completion* line, which is connected to the complement of the dummy wordline is the last signal to switch high. This signals the completion of the evaluation operation. The completion signal of the PLA switches in each cycle. This signal is used to phase lock the PLA delay with the *BCLK* signal.

VII-D. The Adaptive Body Biasing Solution

In this chapter, we propose a technique that uses self-adjusting body bias, to *phase lock* the circuit delay to a *beat clock*. This phase locking is done for a group of spatially localized Programmable Logic Arrays (PLAs). *Therefore, inter and intra-die process variations are tackled dynamically by our approach, making our sub-threshold circuit design approach a viable means of designing extreme low power circuits*.

PLAs are chosen as the structure of choice for circuit implementation since they can be designed such that the delay is constant for all PLA outputs, regardless of the input patterns applied. This eliminates the requirement of coming up with a worst-case delay for

[2]Note that in the sub-threshold region a transistor is either *off* or *less off*. For the sake of simplicity, we say that an NMOS transistor is *on* when its gate is at VDD and *off* when its gate is at GND. Similarly we say a PMOS transistor is *on* when its gate is at GND and *off* when its gate is at VDD.

logic which we would require if the circuit was implemented using standard cells.

In our approach the circuit consists of a multi-level network of interconnected, medium-sized dynamic NOR-NOR PLAs[3]. Spatially localized PLAs are clustered, and each cluster of PLAs shares a common Nbulk node. This Nbulk node is driven by a bulk bias adjustment circuit (one per PLA cluster), whose task it is to synchronize the delay of a representative PLA in the cluster, to a globally distributed *beat clock (BCLK)*. The beat clock is an external signal, derived from the system clock. If the user would like a high speed of operation, they increase the duty cycle of *BCLK*, and all PLAs in our design speed up to synchronize to *BCLK*. Conversely, the user can reduce the duty cycle of *BCLK* (when the computational needs are relaxed), and the PLAs slow down and synchronize to *BCLK* again. In this way, we can implement a synchronous design methodology using sub-threshold PLAs, in a manner that is insensitive to inter and intra-die processing, temperature and voltage variations.

The main problem with a sub-threshold conduction based design approach is the strong dependency of the sub-threshold current I_{ds}^{sub} on process, temperature and voltage variations. We can see from the sub-threshold current equation that I_{ds}^{sub} has an exponential dependence on temperature. Similarly, its dependence on V_{gs} (or in other words, VDD) and process factors such as V_T is also exponential.

We plotted the variation of sub-threshold circuit delay[4] (for a precharged NOR-NOR PLA) against temperature, while varying various process, voltage and temperature parameters. The results are shown in Figure VII.2. The light area represents the envelope of delays with respect to PVT variations *when no compensation was applied*. Note that the

[3]By medium sized PLAs, we mean PLAs that have about 5-15 inputs, 3-8 outputs, and 10-20 rows.

[4]This is defined as the delay from the start of the evaluation phase of the computation, to the time that the *completion* signal has switched

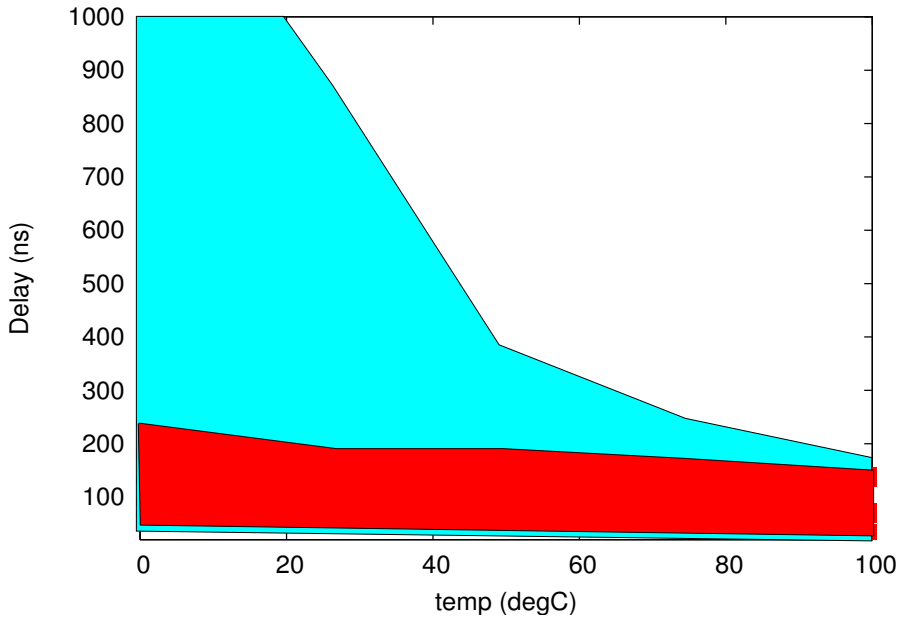


Fig. VII.2. Delay range with and without our dynamic body bias technique

PLA delay varied by an order of magnitude. Further, in the light area of the plot, for very low temperatures (to the top and left of the Figure VII.2) the PLA outputs did not switch at all. The parameters that were varied to compute the envelope were l_{eff} ($\pm$5% variation), V_T ($\pm$5% variation) and VDD ($\pm$10% variation). These variation values represent 3σ variation around the mean, and are obtained from [64]. The dark region of Figure VII.2 represents the PLA delay variation *after* our self-adjusting body bias technique was applied. The same variations were applied as for the light region. Note the significant reduction in the effect of PVT variations on PLA delay. Also, and importantly, *these adjustments are done in a closed-loop manner during circuit operation*. We next describe how these adjustments are made.

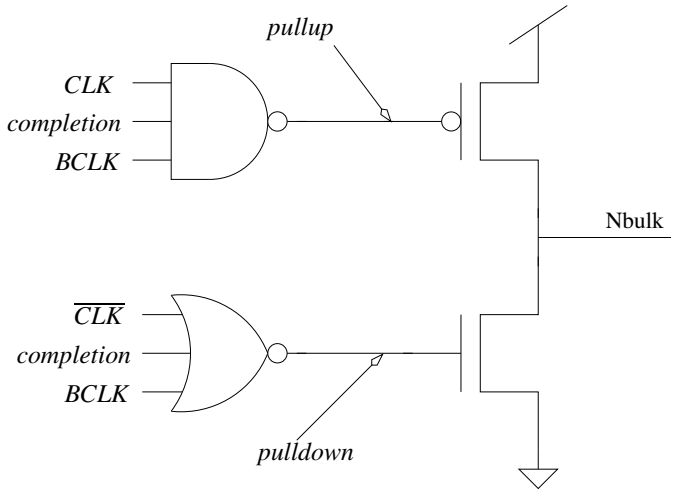


Fig. VII.3. Phase detector and charge pump circuit

VII-D.1. Self-adjusting Bulk-bias Circuit

Our self-adjusting body bias scheme controls the substrate voltage of a cluster of PLAs in a closed-loop fashion, by ensuring that the delay of a representative PLA in the cluster is phase locked to the *BCLK* signal. The phase detector and charge pump circuits for our design are shown in Figure VII.3.

The NAND gate in this figure detects the case when the completion signal is too slow, and generates low-going pulses in such a condition. These pulses are used to turn on the PMOS device of Figure VII.3, and increase the *Nbulk* bias voltage, resulting in a speed-up in the PLA. The waveforms of the signals for this case are shown in Figure VII.4. Similarly, when the completion signal is fast, the NOR gate generates pulses to turn on the NMOS device of Figure VII.3, and hence decrease the *Nbulk* bias voltage. The waveforms, for this situation is shown in Figure VII.5.

Note that in general, *BCLK* is derived from *CLK*, having coincident falling edges with

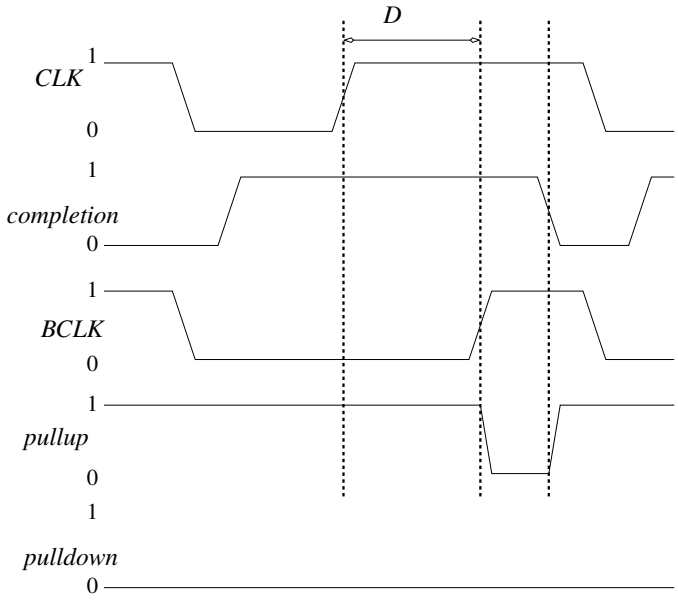


Fig. VII.4. Phase detector waveforms when PLA delay lags *BCLK*

CLK but a rising edge which is delayed by a quantity *D* from the rising edge of *CLK*. This quantity *D* is the delay which we want for the evaluation of all PLAs. The value of *D* is computed by analyzing Figure VII.2. We determine the largest value of delay D_{max} of the PLA for the dark region over temperatures. Now we add a suitable setup delay and phase lock error margin (in our case, we took this to be 20ns) to D_{max} to obtain *D*. Note that a larger margin can be chosen if we would like to be more conservative.

If the completion has not occurred by the time *BCLK* rises, a downward pulse is generated on the *pullup* signal, which forces charge into the Nbulk node, resulting in faster generation of *completion*. Note that at this time, *pulldown*, the signal which is used to bleed off charge from *Nbulk*, is low.

The NOR gate in Figure VII.3 generates high-going pulses to turn on the NMOS transistor when the PLA delay leads *BCLK*. These pulses drive the NMOS device in Fig-

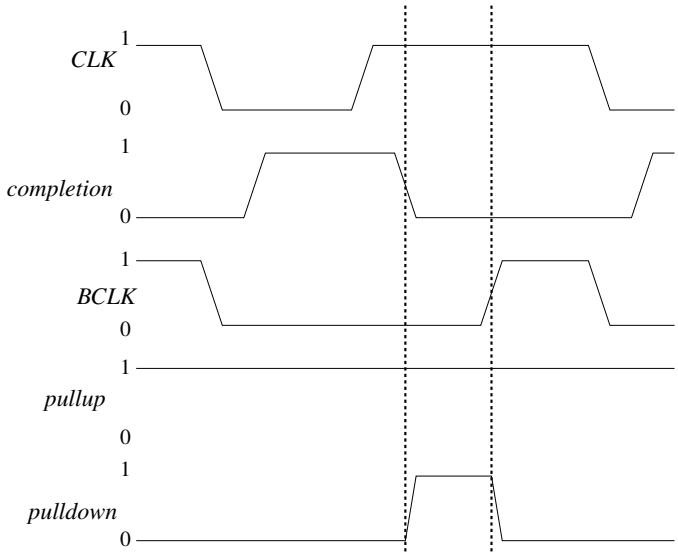


Fig. VII.5. Phase detector waveforms when PLA delay leads *BCLK*

ure VII.3, bleeding charge out of *Nbulk* and thereby slowing the PLA down.

There are several observations we can make about this approach:

- Note that the PLAs in our approach operate *just fast enough* to stay synchronized with *BCLK*, thereby minimizing circuit power for a given speed of operation.
- Note that *BCLK* is used for clocking the memory elements in the design as well as for phase locking the delay of the PLA clusters.
- We do not perform bulk voltage control for PMOS devices, since there are very few PMOS devices per PLA, and they are mostly utilized for pre-charging purposes. It is crucial to perform bulk voltage control for NMOS devices since they are used to perform the computation during the evaluate phase of the clock.
- Sequential designs are implemented using *BCLK* as the system clock (as well as

the clock used to synchronize the delays of the combinational part of the design). Additional margin is included in T_{BCLK}, to account for setup delays of the memory elements and and lock margin. The margin for hold times of the memory elements need not be considered since these elements are latched at the falling edge of $BCLK$.

- The distribution of the power supply and ground signals should be performed using a low-resistance supply distribution methodology such as a layout fabric [65, 66]. The power distribution network in these papers had significantly lower iR drops than existing power distribution approaches (up-to 20$\times$ lower than traditional approaches [65]). The distribution of a sub-threshold VDD signal could be challenging, but this challenge can be averted by using a high quality power distribution grid. Also, the switching currents in the sub-threshold design methodology are up to a couple of orders of magnitude smaller than in traditional designs, alleviating the power supply distribution problem significantly.

- We use PLAs as the circuit implementation structure because we can design them such that the delay of all outputs is constant, regardless of the input vector applied. Hence, the task of finding the critical delay path (which needs to be solved in other bulk bias control approaches such as [62]) is avoided. Also, design methodologies using a network of medium sized PLAs was shown [66] to be a viable way to perform digital design, resulting in improved area and delay for a design. In a standard cell based flow, there is an intervening technology mapping step, which often negates the benefits of technology-independent logic optimization. A network of PLAs on the other hand, allows us to carry forward the benefits of technology-independent multi-level logic synthesis. Finally, a design implemented using such a network of PLAs can be easily mapped into a structured ASIC setting [63].

VII-E. Experimental Results

We implemented our technique using PLAs as described in Section VII-C.1. Each cluster consisted of 1000 spatially localized PLAs. PLAs were designed with 12 inputs, 12 rows and 6 outputs. The layout of each PLA occupied slightly over $25\mu \times 15\mu$, so each cluster was of size 0.8mm $\times$ 0.5mm. We simulated these PLAs using the the 65nm BSIM4 model cards from [27].

Table VII.1. Selecting the value of D

Corner	VDD	V_{Nbulk}	0°C	27°C	50°C	75°C	100°C
SS	0.18	0	n/a	685.24	376.84	251.59	169.46
		max	219.34	167.79	126.52	105.11	86.47
	0.20	0	n/a	866.15	376.12	217.01	156.98
		max	138.25	108.54	91.39	77.71	67.94
	0.22	0	n/a	n/a	360.33	204.91	148.71
		max	92.92	78.64	66.41	59.06	51.45
TT	0.18	0	254.45	168.68	139.63	105.60	82.73
		max	113.69	91.07	76.38	63.76	54.50
	0.20	0	189.59	126.91	100.19	82.22	69.11
		max	78.67	64.48	55.88	47.69	42.12
	0.22	0	135.12	102.17	82.68	63.66	59.77
		max	54.55	45.55	40.52	36.45	37.99
FF	0.18	0	88.45	67.41	61.34	46.91	40.20
		max	60.16	46.56	40.51	34.06	30.68
	0.20	0	65.41	52.19	43.11	37.60	33.48
		max	41.33	33.54	29.76	24.91	23.50
	0.22	0	47.53	40.03	34.03	30.45	25.70
		max	28.68	23.58	22.71	22.33	20.56

Table VII.1 reports the PLA delay as a function of several varying parameters. The delay is expressed as a function of l_{eff} and V_T, with varying VDD and V_{Nbulk}. The notation 'S' indicates a slow corner, 'F' indicates a fast corner, and 'T' represents a typical corner. This table represents the PLA delay range that our active compensation technique can phase lock to the beat clock. Note that a 'n/a' entry in Table VII.1 indicates that for the particular set of parameters, the PLA did not switch at all. The magnitude of variations for l_{eff} and V_T are as described earlier in this chapter, and are obtained from [64]. Note that for any process and VDD entry at any temperature, the highest speed possible is when V_{Nbulk} is maximum (i.e. set to the value of VDD for that simulation). Also, note that the ratio of

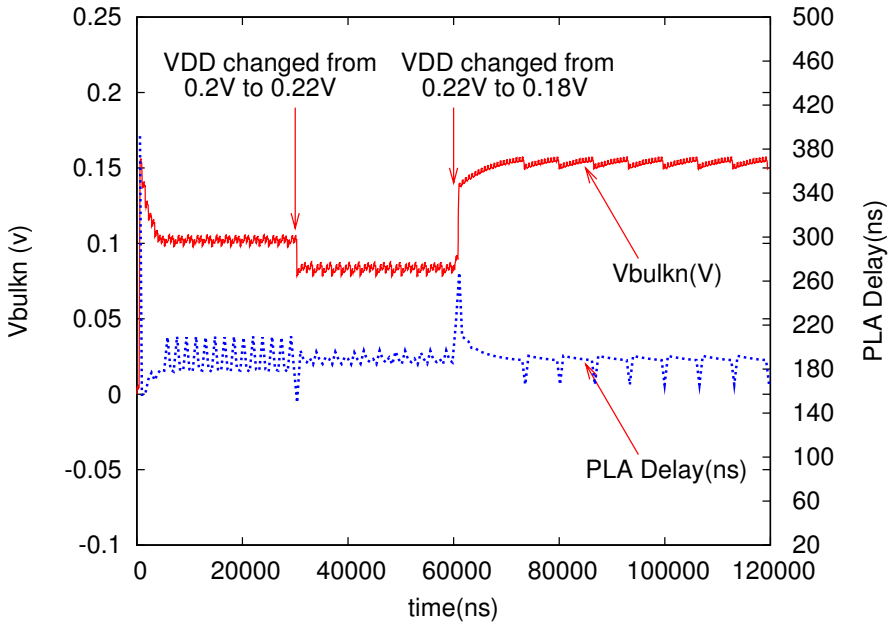


Fig. VII.6. Dynamic adjustment of PLA delay and V_{Nbulk} with VDD variation

the fastest to the slowest delay in this table is as high as 42:1, and our active body bias adjustment can compensate for any of these delay values.

Using Table VII.1, we can find the value of D (the amount by which we delay the rising edge of CLK to obtain $BCLK$ – please see Figure VII.4 for illustrative purposes). We find the largest delay in the table for all rows with maximum V_{Nbulk}, and add a guard-band value to this (to account for lock margin and setup margin for the memory elements). This quantity is the value of D used.

When we utilize our approach using self-adaptive body bias, the process variations described above are reduced to the dark region in Figure VII.2. In other words, our approach is able to work for all the conditions in Table VII.1, with a delay contained in the darkened region in Figure VII.2. The PLA delays for our approach are very tightly bounded across all these operating conditions.

Figure VII.6 describes a SPICE [32] plot of the variation of bulk voltage and PLA

delay in our self-adjusting bulk bias scheme. The (higher) solid line represents the value of V_{Nbulk}, while the (lower) dotted line represents the PLA delay. Note that in this figure, the VDD value was initially 0.2V. At time 30,000ns, VDD was changed to 0.22V. Note that in response to this change, our body bias adjustment circuitry modified V_{Nbulk} to a lower value in order to slow the PLAs down. At time 60,000ns, the VDD value was changed to 0.18V, and consequently, our bias adjustment circuit modified V_{Nbulk} to a higher value to speed up the PLAs and keep them phase locked with $BCLK$. Note that in spite of all the changes in VDD, the delay of the PLA stays tightly bounded. This simulation was done for a slow corner, at 27°C.

VII-F. Loop Gain of the Adaptive Body Biasing Loop

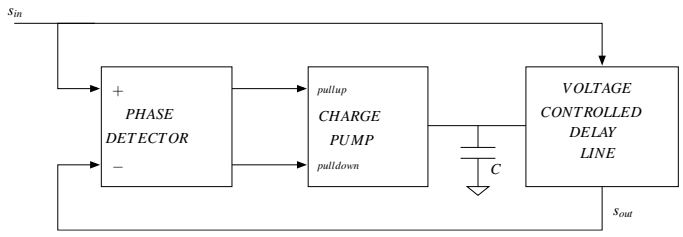


Fig. VII.7. Example of a traditional charge-pump DLL (adapted from [3])

In our scheme, we "phase lock" the delay of a representative PLA to a beat clock. We use a charge-pump to adjust the body bias voltage of the PLA which in turn controls the delay of the PLA. In principle, this scheme is a charge-pump Delay Locked Loop (DLL). An example of a traditional charge-pump DLL is shown is Figure VII.7. In our case the representative PLA whose delay we phase lock to the beat clock takes the place of the Voltage controlled delay line (VCDL) in Figure VII.7. The phase-detector and charge-

pump are as shown in Figure VII.3 The signals s_{in} and s_{out} refer to the input clock signal (or beat clock signal) and the PLA completion signal respectively.

Based on the model shown in Figure VII.7, we can derive the following expressions [3]:

$$s_{out}(n) = s_{in}(n-1) - K_{PLA}V_C(n) \tag{7.2}$$

$$V_C(n) = \frac{s_{in}(n-1) - s_{out}(n-1)}{C} I_p T \tag{7.3}$$

where, $V_C(n)$ is the control voltage (body-biasing voltage) applied at the n^{th} clock cycle, K_{PLA} is the delay gain of the PLA (ds_{out}/dV_C), I_p is the current that the charge-pump can deliver to pullup or pulldown the control node (Nbulk node) and T is the time period of the clock. The physical meaning of Equation 7.2 is that the arrival time of the completion signal of the representative PLA at clock cycle n is the dependent on the arrival time of the beat clock at the $n-1^{th}$ clock cycle, the delay gain of the PLA and the control voltage at the n^{th} clock cycle. Equation 7.3, merely states that the control voltage at the n^{th} clock cycle is dependent on the beat clock and PLA delay at the $(n-1)^{th}$ clock cycle, the capacitance C of the control node, the time period T and the rate at which the charge-pump can pull-up and pulldown the control node.

The delay of the PLA is dependent on (inversely proportional to) the operating currents, in our case sub-threshold leakage currents. Hence the delay of the PLA (D_{PLA}) can be written as

$$D_{PLA} = \frac{k_1}{I_{ds}} \tag{7.4}$$

In the sub-threshold region

$$I_{ds} = \frac{W}{L} I_{D0} e^{(\frac{V_{gs}-V_T-V_{off}}{nv_t})} [1 - e^{-\frac{V_{ds}}{v_t}}] \tag{7.5}$$

We are only concerned with the change in I_{ds} due to change in the body-bias voltage. Hence

the expression for I_{ds} can be reduced to:

$$I_{ds} = k_2 e^{(\frac{V_{gs}-V_T-V_{off}}{nv_t})} \tag{7.6}$$

The body effect equation is as follows:

$$V_T = V_T^0 + \gamma(\sqrt{|(-2)\phi_F + V_{sb}|} - \sqrt{|2\phi_F|}) \tag{7.7}$$

In the above expression for V_T,

$V_{sb} = 0 - V_C$

since the source terminal is tied to GND and the bulk terminal is the control node. Substituting the above expression for V_{sb} and the expression for V_T (Equation 7.7) in the expression for I_{ds} (Equation 7.6) we get:

$$I_{ds} = k_3 e^{(\frac{-\gamma(\sqrt{|(-2)\phi_F - V_C|})}{nv_t})} \tag{7.8}$$

Substituting the above expression for I_{ds} in Equation 7.4 we get:

$$D_{PLA} = k_4 e^{(\frac{\gamma(\sqrt{|(-2)\phi_F - V_C|})}{nv_t})} \tag{7.9}$$

Differentiating Equation 7.9 with respect to V_C we get:

$$K_{PLA} = \frac{dD_{PLA}}{dV_C} = k_5 \frac{e^{(\frac{\gamma(\sqrt{|(-2)\phi_F - V_C|})}{nv_t})}}{\sqrt{|(-2)\phi_F - V_C|}} \tag{7.10}$$

The expression for $s_{out}(n)$ from Equation 7.2 can be re-written (as was shown in [3]) as:

$$s_{out}(n) = s_{in}(n-1) - K_{loop}[s_{in}(n-1) - s_{out}(n-1)] \tag{7.11}$$

Here K_{loop} is the loop gain given by:

$$K_{loop} = \frac{K_{PLA} I_p T}{C} \tag{7.12}$$

In the expression for loop gain K_{loop}, the current I_p is proportional to the width W of the pullup or pulldown device. Hence, from Equations 7.12 and 7.10 we get the expression for loop gain to be:

$$K_{loop} = k_6 \frac{WTe^{(\frac{\gamma(\sqrt{|(-2)\phi_F - V_C|})}{nv_t})}}{C\sqrt{|(-2)\phi_F - V_C|}} \tag{7.13}$$

The loop gain is hence proportional to the drive strength of the charge-pump and inversely proportional to the capacitance of the control node. The response of our closed-loop adaptive body-biasing scheme can be adjusted using these two parameters.

VII-G. Chapter Summary

Sub-threshold circuits demonstrate a dramatically reduced power consumption compared to the traditional design approaches. They are however extremely sensitive to PVT variations. In this chapter we presented a practical sub-threshold design methodology, which actively compensates for variations in supply, temperature and process. The power of our approach is its ability to adapt to inter and intra-die PVT variations, enabling a significant yield improvement.

In our design methodology, we propose using a multi-level network of medium sized Programmable Logic Arrays (PLAs) as the circuit implementation structure. Spatially localized PLAs are grouped into clusters which share a common Nbulk terminal. The design uses a global *beat clock* to which the delay of a representative PLA in this spatially localized cluster, is ”phase locked”. Based on whether the delay of a representative PLA in any cluster leads or lags the beat clock, our approach either automatically decreases or increases the NMOS transistor bulk voltage for the cluster of PLAs. The synchronization is performed in a closed-loop fashion, using a phase detector and a charge pump which drives the bulk nodes of the PLAs in the cluster. Our results demonstrate that our technique is able to dynamically phase lock the PLA delays to the beat clock across a wide range of

PVT variations. Our adaptive body-biasing scheme, is in principle a charge-pump DLL. We analyzed our scheme and derived the loop gain of the system. We find that the response of the system can be tuned by adjusting the drive strength of the devices in the charge-pump and the capacitance of the control (Nbulk) node.

CHAPTER VIII

OPTIMUM VDD FOR MINIMUM ENERGY

VIII-A. Chapter Overview

While sub-threshold circuit design approaches can reduce the power consumption significantly, a design operating in the sub-threshold region is not necessarily optimal in terms of energy consumption. In this chapter, we describe a technique to find the energy optimum VDD value for a design, and show that for minimum energy consumption, the circuit may need to be operated at VDD values which are slightly higher than the NMOS threshold voltage value. We study this problem in the context of designing a circuit using a network of dynamic NOR-NOR PLAs.

VIII-B. Introduction

Power is minimized by operating the design at a lower voltage. However, a practical approach to designing VLSI ICs with minimum *energy* consumption would be very desirable for a large and growing class of practical applications. While it has been shown that power consumption is lower for lower voltages, the energy consumption per operation (i.e. the energy consumption for a logic gate to perform one computation) is not necessarily lower for lower VDDs. This is due to the fact that since switching times are longer, the power consumption over that longer switching period causes a greater energy consumption. In this chapter we describe an approach to finding the optimal VDD value for energy minimization. We assume that the circuits in question can be operated over a range of VDD values (including sub-threshold and super-threshold values of VDD).

We address the problem of finding the optimal VDD value for minimum energy consumption in a design scenario where a design is implemented using a network of medium

sized Programmable Logic Arrays (PLAs) [65]. This design approach was shown recently to be suitable for implementing structured ASICs with a low-NRE cost [63]. Also, it was indicated in a recent keynote talk [67] that PLAs are strong contenders as the circuit implementation structures of choice in future designs.

VIII-C. Related Previous Work

There has been some recent research in the area of sub-threshold operation [61, 52, 58, 59, 60] for standard-cell based designs. These designs consume extremely low power. However, as has been pointed out in [68, 69, 70], while the optimum VDD for minimum power is the lowest possible VDD value, the optimum VDD for minimum energy can be higher, especially in situations when the static power consumption is comparable to the dynamic power consumption.

In [68], a first-order model of the energy-delay product (EDP) is reported . Using this model, the authors find the optimum VDD and body bias point for CMOS circuits operating in strong inversion. In [70], the authors examine the effects of device sizing on energy for standard-cell based circuits operating in the sub-threshold region. In [69], the performance and energy dissipation contours for CMOS circuits operating in the sub-threshold region are presented, to help find the optimum VDD and threshold voltage. The authors of [69] also point out that these contours change depending on the switching probabilities of the circuit nodes. Hence the optimum VDD is heavily dependent on the type of circuit. Similarly, in [71], the authors describe theoretical and practical considerations for energy minimization in dynamic voltage scaled systems, allowing for sub-threshold operation.

In this work, as in [69, 71], we attempt to find the optimum VDD which minimizes energy for a circuit. However, in contrast to prior approaches, we use fixed-size dynamic NOR-NOR PLAs instead of standard-cells as the circuit implementation approach. One of

the advantages of this design choice is that it allows us to come up with the optimum VDD for any design with just the knowledge of the logic depth (in terms of the number of PLAs) of the design, and the energy characterization data of a *single* PLA. This is not feasible for previous standard cell based approaches. As a consequence, our approach is applicable to network of PLA based designs, including structured ASICs [63] implemented under this methodology. Further, in contrast to the approaches of [69, 71], our network of PLA based approach has an energy consumption which is *highly predictable, and largely independent of the input vector applied to the design*. This fact arises from the regularity inherent in the PLAs. Also, in contrast with [71], we study the dependence of the optimal VDD point on temperature.

The ability to find the optimum VDD for a network of PLA circuit using the characterization data from just a single PLA, allows us a significant advantage in a practical design setting. We can find the optimum VDD for a circuit by only knowing its topological depth in terms of number of PLAs. We do not need to know any additional design details.

VIII-D. Preliminaries

The aim of this work is to explore how energy can be minimized in a circuit designed using a network of precharged NOR-NOR PLAs. Towards this end, we first explore the effect (in terms of power, delay and energy consumption) of changing VDD and Vbulkn (the body bias of NMOS devices in the PLA) for a single PLA and then use this information to help find an optimum VDD value for a circuit designed using these PLAs.

The PLA we use is a precharged NOR-NOR PLA (similar to the ones used in [65, 72, 73, 74] and Chapter VII). The schematic view of the PLA circuit is shown in Figure VII.1 in Chapter VII. The operation of these PLAs is also explained in Section VII-C. The PLAs

we consider have a fixed number of inputs (12), outputs (6) and rows (12)[1].

A circuit implemented using a network of PLAs operates as follows. All PLAs precharge when the global clock signal is low. When the global clock is high, the PLAs evaluate. The evaluation condition of a PLA of topological depth i is the global clock, gated by the completion signal of the slowest PLA among the PLAs of level $i-1$.

VIII-D.1. Some Definitions

Since the PLAs used are of fixed size, the characterization of a single PLA provides enough information to estimate the delay, power and energy consumption of a circuit built using these PLAs as building blocks. The regularity of the PLAs, which allows us to infer circuit level delay, power and energy estimates from those of a single PLA, is an additional advantage of this design approach.

We divide the modes of operation of the PLA into 4 different phases in order to characterize it more easily. These are the *Precharging mode*, the *Precharged mode*, the *Evaluating mode* and the *Evaluated mode*. This partitioning of modes is shown in Figure VII.1. The *Precharging mode* refers to the period of operation during which the PLA is precharging. In this mode, all wordlines and output lines get pulled high. The *Precharging time*, T_{pchg} is defined to be the time from which the clock starts to go low (1% below VDD) to the time when the completion signal of the PLA reaches logic high (within 1% of VDD). Similarly the *Evaluating mode* refers to the period when the PLA is evaluating. This is the period during which the wordlines and the output lines are switching low (depending on the inputs to the PLA). The *Evaluating time*, T_{eval} is defined to be the time from when the clock starts to go high (1% of VDD above GND) to the time when the completion line reaches logic low (reaches within 1% of VDD above GND). The *Precharged mode* refers

[1]We fix these values for each PLA in the design so as to be able to utilize the PLAs in a structured ASIC setting, allowing for a low-NRE design approach.

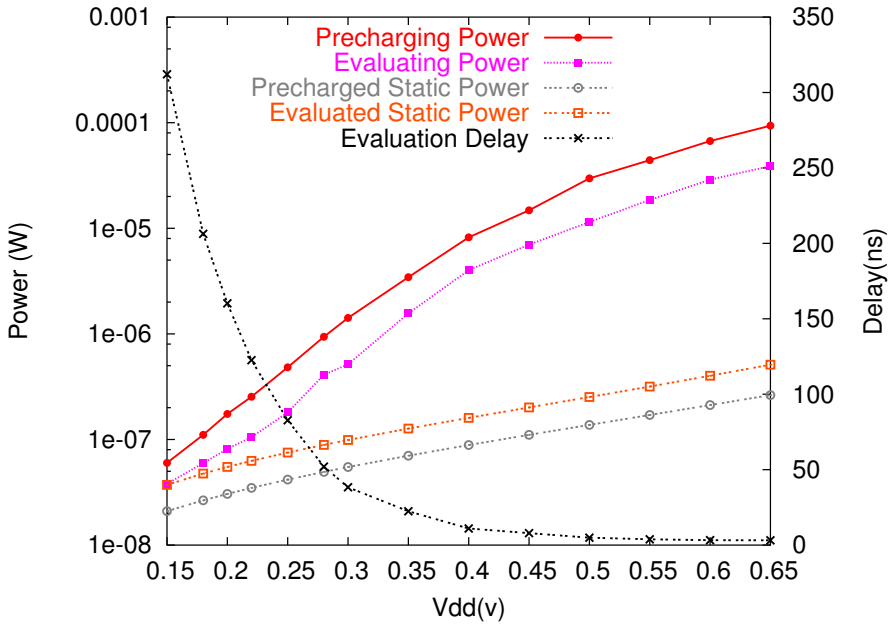


Fig. VIII.1. Power dissipated, delay in the four modes with varying VDD (Vbulkn = 0V)

to the period when the PLA is precharged and is idle (waiting for the clock to go high to start evaluation). Similarly, the *Evaluated mode* refers to the mode of operation where the PLA has completed evaluation and is idle (waiting for the clock line to go low to start the next precharge operation). The power consumed in the Precharging and the Evaluating modes is classified as *dynamic power consumption*, while the power consumption in the Precharged mode and the Evaluated mode is classified as *static power consumption*. Note that the static power consumption includes power consumption due to all forms of leakage currents (sub-threshold leakage, gate leakage and gate induced drain leakage (GIDL)). Let $EvalEnergy_{dyn}$ denote the energy consumption in the Evaluating mode, $PchgEnergy_{dyn}$ denote the energy consumption in the Precharging mode, $EvalPwr_{sta}$ denote the power dissipated in the Evaluated mode and $PchgPwr_{sta}$ denote the power dissipated in the Precharged mode. The *evaluation delay* is defined as the difference between the time instant the clock line voltage crosses VDD/2 (clock line rising) and the instant when the completion line

crosses VDD/2 (completion line falling). In the operation of the PLA, the evaluation delay is the critical delay of the PLA.

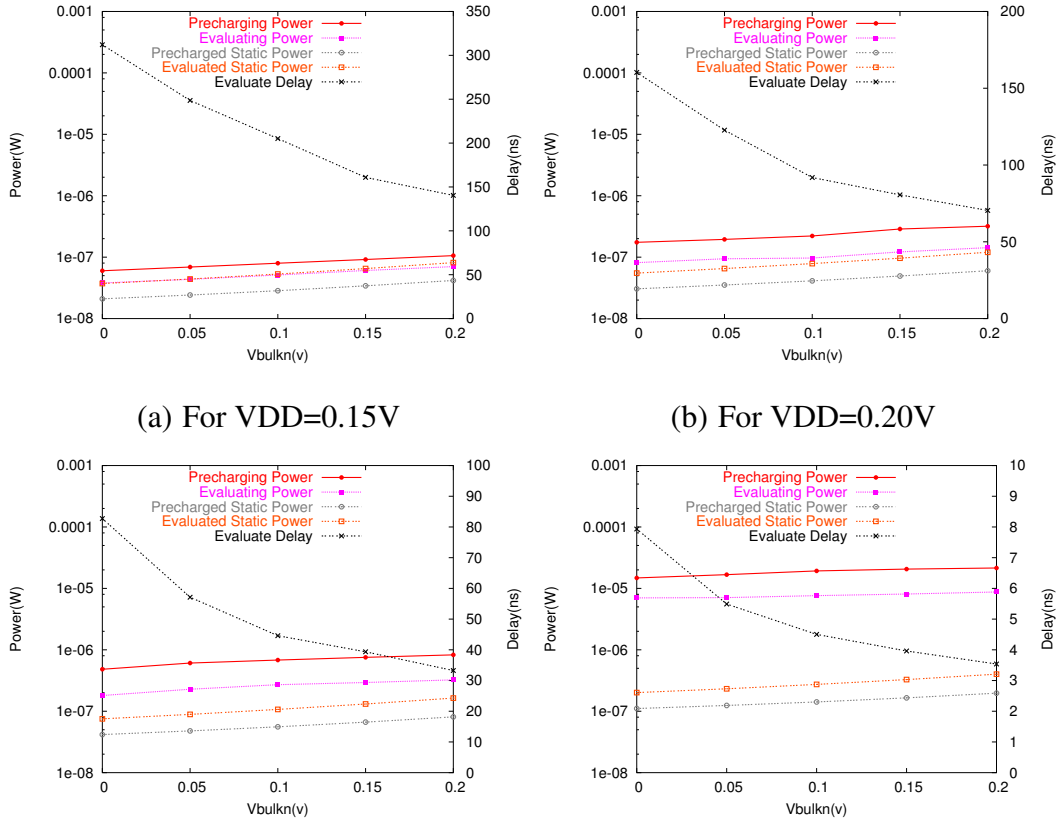
VIII-E. Experiments

For our simulations, we used Spice3 [32] with 65nm BSIM4 [27] model cards. The threshold voltages for our devices were $V_{T_n} = 0.22V$ and $V_{T_p} = -0.22V$. In this section we will discuss the results of these simulations and describe a methodology to find an optimum VDD value for a circuit, so as to minimize energy consumption. The range of VDD values that are of interest vary from slightly below V_T to a few 100mV above V_T. Hence, we refer to our operating voltage range as *near-threshold*.

Figure VIII.1 shows the plot of power for the PLA (for each of the four modes) for an operating temperature of 25°C. The power is plotted at varying VDD levels. The plot also shows the dependence of the evaluation delay on VDD. Not surprisingly, the delay increases at lower voltages while power dissipation is reduced. Similar results were seen at other temperatures and different Vbulkn values.

Figure VIII.2 shows plots of the power dissipated for the different modes with varying Vbulkn at different VDD values. The temperature was fixed at 25°C. The plots for other temperatures are similar. The evaluation delay variation with Vbulkn is also shown. As can be seen from these plots, at low voltages (especially at sub-threshold voltages), a forward body bias of 0.2V can give more than a $2\times$ speedup but with a proportionate power penalty. Forward body biasing helps reduce delay for higher voltages as well, but the effect is greater at low / sub-threshold voltages.

Figure VIII.3 shows plots of the energy consumption with varying Vbulkn for different VDD values at a temperature of 25°C. These plots indicate that even with the increase in power due to forward body biasing, the energy consumption doesn't increase significantly



(a) For VDD=0.15V

(b) For VDD=0.20V

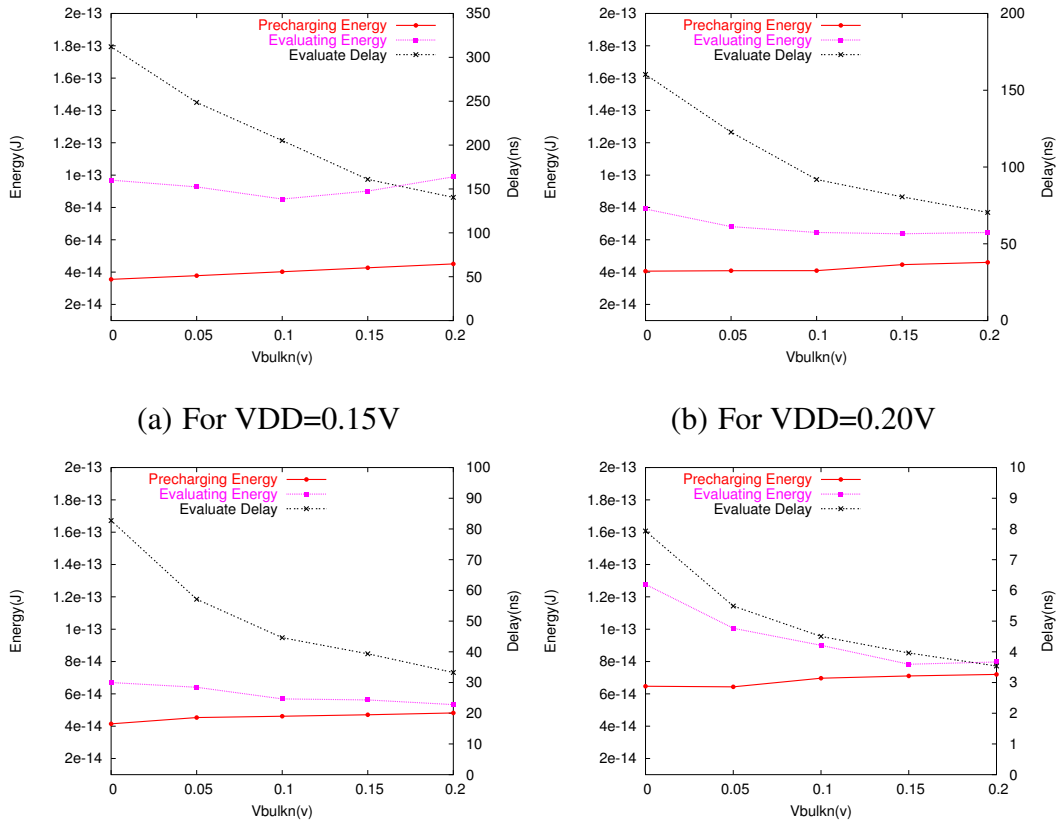
(c) For VDD=0.25V

(d) For VDD=0.45V

Fig. VIII.2. Power and delay in all four modes with varying Vbulkn

and can in fact decrease with increasing forward body bias. This would suggest that a forward body bias helps since it decreases delay without an energy penalty. However, rather than drive this body bias voltage with a fixed value, it is suggested that this body-bias control be used adaptively as suggested in Chapter VII to control the speed of the PLA circuit over varying process corners and temperatures. This is because devices in the sub-threshold region of operation are more susceptible to temperature and process variations.

Figure VIII.4 plots the energy consumption in the evaluating period and in the precharging period of the PLA. The evaluation delay is also shown. Note that the evaluation delay is measured at the VDD/2 crossing points. This delay is smaller than the evaluating time



(a) For VDD=0.15V

(b) For VDD=0.20V

(c) For VDD=0.25V

(d) For VDD=0.45V

Fig. VIII.3. Energy consumption and delay in the two dynamic modes, with varying Vbulkn

T_{eval} (see definitions in Section VIII-D.1).

Intuitively, for minimum energy consumption, no time should be spent in the idle modes (Precharged mode and Evaluated mode). However, in a circuit constructed using a network of PLAs of fixed size, some of the PLAs may have to remain in the Precharged state or in the Evaluated state for a certain period of time. This duration is dependent on the topological depth of the network of PLA circuit (as we shall see in Section VIII-E.1).

The evaluation energy consumption is plotted against VDD in Figure VIII.5. The different curves denote the different ratios of evaluating time to time spent in the evaluated state. T0 represents only the evaluating energy consumption (no time spent and hence

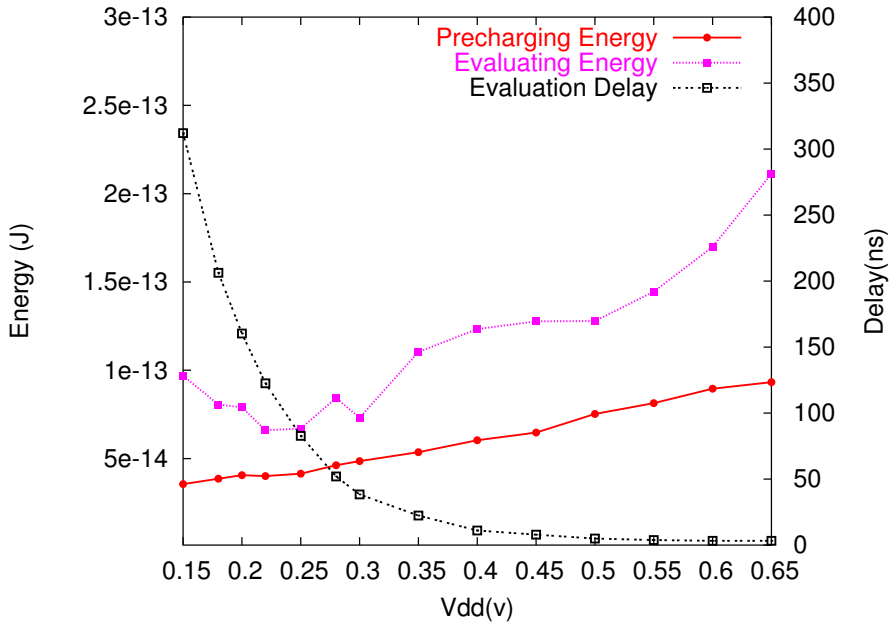


Fig. VIII.4. Energy consumption, delay in the two dynamic modes with varying VDD (Vbulkn = 0V)

no energy consumed in the evaluated state). T1 denotes the sum of energy consumption during the evaluating period (dynamic energy consumption in the evaluating period) and energy consumption in the evaluated state for a period equal to the evaluating time. In other words, the curve T1 plots $energy = EvalEnergy_{dyn} + (T_{eval} \times EvalPwr_{sta})$. Similarly the curve T2 plots $energy = EvalEnergy_{dyn} + (2 \times T_{eval} \times EvalPwr_{sta})$, and so on. In essence, Figure VIII.5 plots the energy consumption for different activity factors i.e. the ratios of time spent in the evaluating state to the time spent in the static (idle) evaluated state.

As can be seen from the plot in Figure VIII.5, as more time is spent in the static (idle) modes, (i.e. in regions where static power is dissipated), the optimum VDD value (which minimizes energy) tends to shift to higher values.

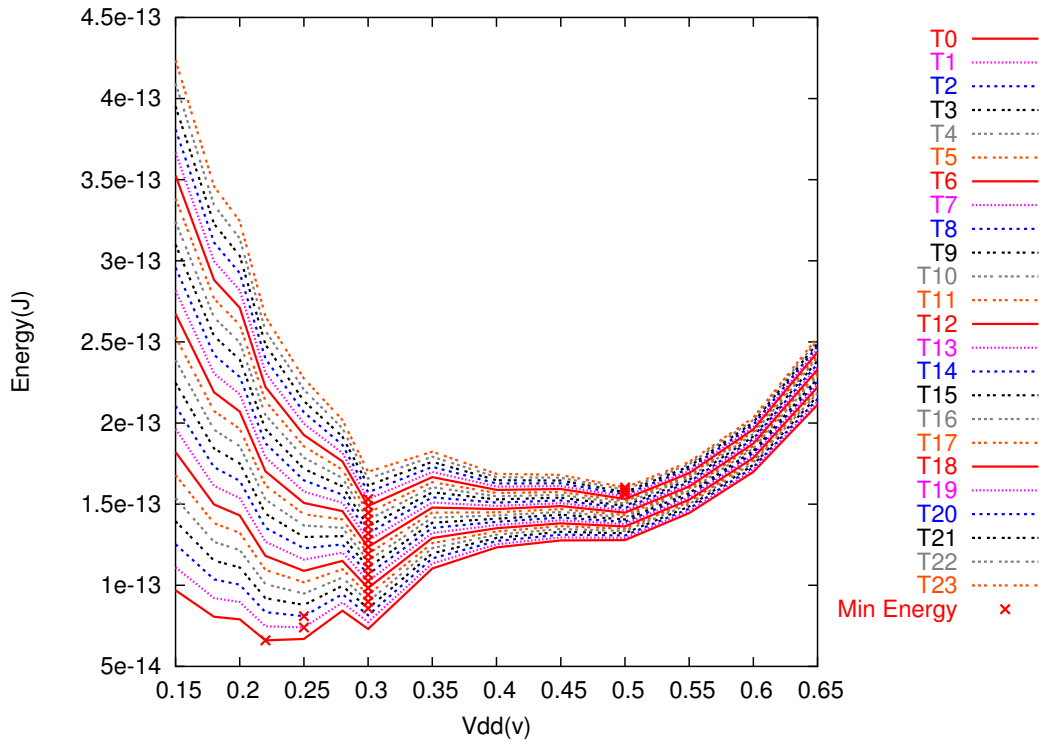


Fig. VIII.5. Energy consumption over different activity factors (Vbulkn = 0V)

VIII-E.1. Energy Estimation for a Circuit of PLAs

The operation of a combinational circuit designed with a network of multi-level fixed size PLAs is as follows. Assume the circuit has a topological depth D. In other words, the longest path between any circuit input and any circuit output traverses D PLAs. All PLAs are precharged simultaneously. Once all the PLAs are precharged, the global clock line goes high for all the PLAs. The PLAs evaluate in a domino fashion, starting with PLAs of topological level 1 and proceeding to PLAs of topological level D. The local clock of the level 1 PLAs is ungated, so level 1 PLAs evaluate as soon as the global clock goes high. The local clock of level i PLAs is gated by the completion signal of a representative level $i-1$ PLA. As a result, once the *completion* signal of level $i-1$ PLAs goes low, level i PLAs begin evaluation. In this manner, the evaluation of PLAs proceeds in topological

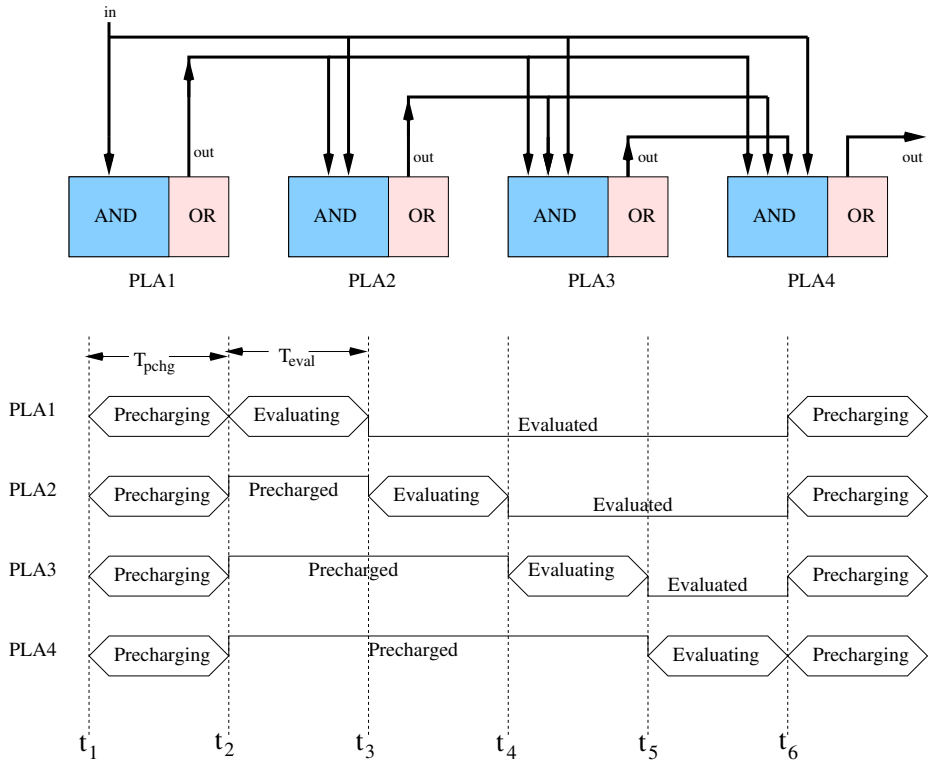


Fig. VIII.6. Circuit built as a series of four PLAs

levelization order.

An example of such a series of four PLAs is shown in Figure VIII.6. PLA1 receives its input externally. PLA2 may receive its inputs externally and/or from PLA1. PLA3 may receive its inputs externally, from PLA2 and/or from PLA1. PLA4 may receive its inputs externally, from PLA3, and/or from PLA2 and PLA1. Note that since the PLAs are of fixed size, each of the PLAs have the same evaluating time.

All four PLAs are precharged at the same time. This operation is completed in time T_{pchg}. Next PLA1 evaluates, taking time T_{eval} to do so. Once the outputs of PLA1 are ready, the next PLA, PLA2 evaluates. Once PLA2 completes evaluation, PLA3 starts evaluating and after PLA3 completes evaluation, PLA4 evaluates. After PLA4 has completed

its evaluation, the circuit is again precharged to get ready for the next set of inputs. As can be seen from the timing diagram in Figure VIII.6, PLA1 is in the evaluated state for a period $t_6 - t_3 = 3 \cdot T_{eval}$. During this period, the energy consumption by PLA1 = $EvalPwr_{sta} \times 3 \cdot T_{eval}$, since the energy consumption during this period is due to the static power consumption in the Evaluated state. Similarly, we find that PLA2 is in the evaluated state for a period = $2 \cdot T_{eval}$, while PLA3 is in the evaluated state for a period = T_{eval}. Figure VIII.6 also reveals that PLA4 is in the Precharged state for the period $t_5 - t_2 = 3 \cdot T_{eval}$ and during this period the energy consumption is given by $PchgPwr_{sta} \times 3 \cdot T_{eval}$ since it is the static power consumption in the Precharged state that contributes to the energy consumption during this time. Similarly, we find that PLA3 and PLA2 are in the precharged state for the durations of $2 \cdot T_{eval}$ and T_{eval} respectively.

Hence, for a PLA in a circuit of topological depth D (in terms of number of PLAs), we can estimate the energy consumption for a PLA at depth k as follows:

$$Energy = PchgEnergy_{dyn} + EvalEnergy_{dyn} + [PchgPwr_{sta} \times T_{eval} \times (k-1)] + [EvalPwr_{sta} \times T_{eval} \times (D-k)] \quad (1)$$

If the circuit consists of n PLAs connected in a chain as in Figure VIII.6, the total energy consumption for all n PLAs is given by:

$$Energy = [(PchgEnergy_{dyn} + EvalEnergy_{dyn}] \times D) + [(D \times (D-1)/2) \times (EvalPwr_{sta} + PchgPwr_{sta}) \times T_{eval}]$$

If the network of PLAs is not structured like a chain, the total energy is computed by summing the energies for each PLA, from Equation 1.

Using this equation, we plotted the energy consumption for network of PLA circuits with different topological depths, with varying VDD. This plot is shown for two different temperatures for circuits up to a logic depth of 24 (labeled Depth0 through Depth23) in Figures VIII.7 and VIII.8.

We find that while power is lower at lower voltages, there is greater energy consump-

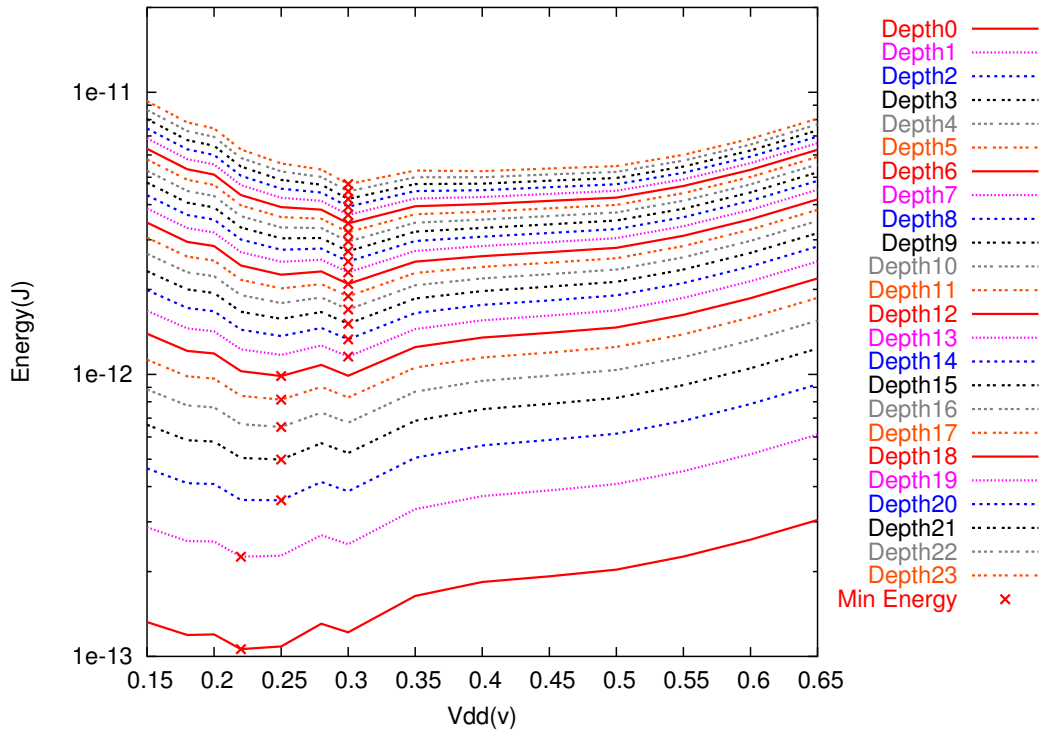


Fig. VIII.7. Total energy consumption per cycle for different logic depths at 25°C (Vbulkn = 0V)

tion per cycle of operation at very low voltages, since the PLA takes longer to switch. This gets worse when the PLA is idle for longer periods (which is inevitable in PLAs circuits with large topological depths). In fact, we find that for such circuits, a higher VDD gives better energy consumption per cycle. Also, we have experimentally validated that the optimum VDD selection is independent of the logic function being implemented, provided the topological depth remains unchanged. Another observation that can be made is that as leakage becomes a larger component of the total power dissipation, the optimum VDD value also increases (in order to reduce the idle time of each PLA). Hence under a forward body bias voltage (which would decrease V_T and thereby increase leakage), the optimum VDD increases.

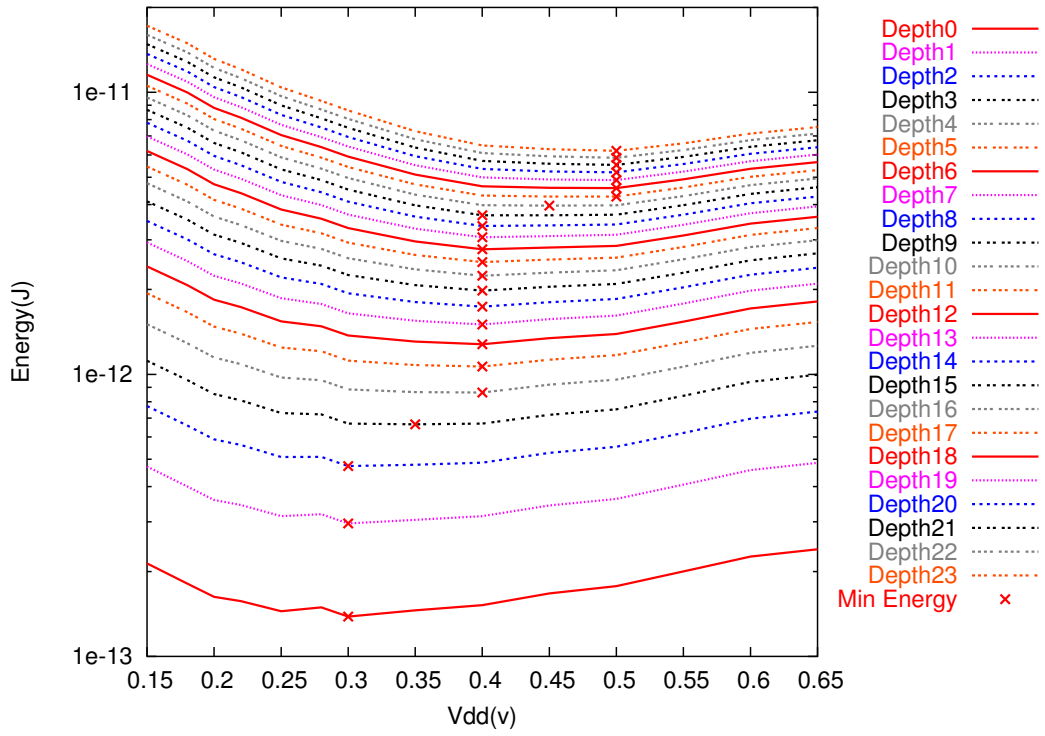


Fig. VIII.8. Total energy consumption per cycle for different logic depths at 100°C (Vbulkn = 0V)

The optimal value of VDD for minimum energy is between V_T and about 1.5·V_T for low temperature operation, while it increases to between $1.5 \cdot V_T$ and $2.5V_T$ for higher temperatures. This suggests that for extreme low power applications such as sensor networks, where the ambient temperature conditions may vary significantly, special temperature compensation circuitry would be required.

VIII-F. Chapter Summary

In recent times, there has been a significant growth in applications for battery powered portable electronics, as well as low power sensor networks. For such systems, energy minimization is a dominant design constraint, whereas circuit speed is a secondary requirement.

In this chapter, we focussed on finding the optimal VDD value for energy minimization of circuits that are implemented in a network of PLA design approach. We find that the optimal VDD value for such designs is close to V_T for circuits with low topological depth, but increases to about $2.5 \cdot V_T$ for circuits with large topological depth and increasing temperature.

CHAPTER IX

RECLAIMING SPEED THROUGH MICRO-PIPELINING

IX-A. Chapter Overview

Sub-threshold circuit design is an appealing means to dramatically reduce power consumption. However, sub-threshold designs suffer from the drawback of being significantly slower than traditional designs. To reduce the speed gap between sub-threshold and traditional designs, we propose a sub-threshold circuit design approach based on asynchronous micropipelining of a levelized network of PLAs. We describe the handshaking protocol, circuit design and logic synthesis issues in this context. Our preliminary results demonstrate that *by using our approach, a design can be sped up by about 7*$\times$, with an area penalty of 47%. Further, our approach yields an energy improvement of about 4$\times$, compared to a traditional network of PLA design. Our approach is quite general, and can be applied to traditional circuits as well.

The key contribution of this work is to come up with a technique which enjoys an extreme low power consumption due to the use of sub-threshold circuitry, but at the same time, compensates for the sub-threshold delay penalty. Such techniques would widen the applicability of sub-threshold circuit design approaches to a broader class of applications. *The proposed approach utilizes a network of PLA (NPLA) based sub-threshold circuit design approach, configured in an asynchronous micropipelined structure to enhance the speed of the circuit.* We provide details about our micropipelined PLA based asynchronous protocol, the logic synthesis approach to decompose a circuit into this circuit paradigm, and also the delay, area, power and energy characteristics of designs which are implemented using our approach.

Sub-threshold circuit design has so far been used in only simple digital circuits and

analog circuits. The design methodologies used in implementing such circuits are ad-hoc. *Our approach provides a systematic EDA framework for the design of complex digital systems using sub-threshold NPLA circuits.* It additionally utilizes an asynchronous micropipelining approach to speed up the sub-threshold design. Our experiments indicate that this approach yields a significant circuit speedup and improvement in energy consumption compared to traditional NPLA designs. Circuit speedup is measured in terms of computational throughput.

IX-B. Our Approach

Our approach to enhancing the speed of sub-threshold circuits is based on implementing the circuit using a micropipelined asynchronous network of PLAs. This implementation has the advantage of increasing the throughput of the circuit to a constant, regardless of the topological depth of the circuit. PLAs with adjacent topological depths in this structure communicate via an asynchronous handshake, which ensures correct operation of the design.

In Section IX-B.1, we describe the operation of the asynchronous micro-pipeline, along with its handshaking protocol. Section IX-B.2 indicates our approach for synthesizing a network of PLAs from a multi-level logic circuit, in a manner which is optimized for an asynchronous micro-pipeline based implementation. We point out that in addition to PLAs, this methodology requires a specialized circuit block (which we call a *stutter* block) which delays signals that traverse multiple levels in the NPLA. Section IX-B.3 describes the design of a single PLA in this methodology, and the handshaking logic within each PLA. We also discuss details of each PLA (maximum number of inputs, outputs and rows) used in our approach. We also describe the design of the stutter blocks used in our approach.

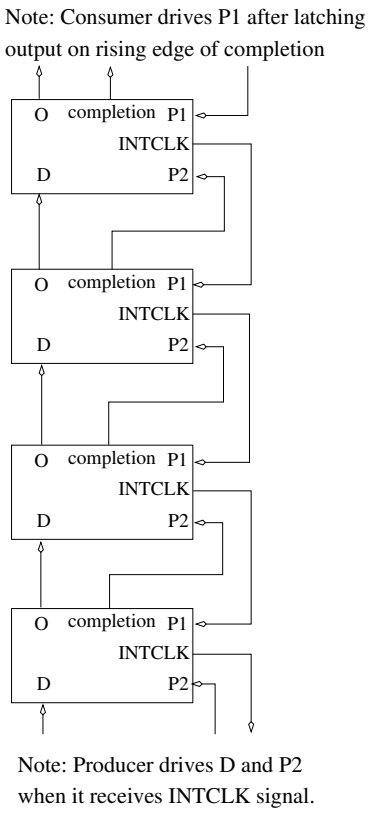


Fig. IX.1. NPLA based asynchronous micropipelined circuit

IX-B.1. Asynchronous Micropipelined NPLAs

Our asynchronous micropipelined design methodology is based on the use of NPLAs [75, 66]. The choice of PLAs for the implementation of the underlying logic is that these structures can be designed to have a constant output delay across all possible input combinations. Also, the use of pre-charged NOR-NOR PLAs results in a compact and fast circuit. It was shown that for a single PLA, the delay was about 48% and the area about 46% compared to a standard cell based design [66], as long as the PLA was medium-sized (with 7-15 inputs, 5-10 outputs and 15-30 rows).

For a robust asynchronous micropipelined implementation, it is critical that the delays of the underlying circuit blocks are extremely predictable. The constant delay of a

dynamic PLA over all input combinations makes it a very attractive choice in this context. Also, we utilize PLAs of fixed size in our approach. In this way we satisfy this important requirement of predictable delay. Note that in a sub-threshold design methodology, circuit delays vary significantly as a function of process, temperature and voltage (PVT) variations, as indicated in Section VI. However, we propose to use an on-the-fly, dynamically delay-compensated NPLA structure, which was shown (in Chapter VII) to dramatically reduce this variation. The residual variation in NPLA delay after applying this technique is minimal. Therefore, a simple guard-banding can achieve a predictable PLA delay across PVT variations in a sub-threshold context.

The structure of the asynchronous micropipelined NPLA is shown in Figure IX.1. Each PLA is a precharged NOR-NOR structure. However, the determination of when a PLA precharges and evaluates is made based on the handshaking protocol. There is no global clock signal in the design. Each PLA has a *completion* signal (which is assumed to switch *high* when evaluation of the PLA completes), which indicates that its outputs have been computed. In Figure IX.1, the inputs of a PLA are indicated as D and the PLA outputs are marked as O. Each PLA has two inputs $P1$ and $P2$ which control the asynchronous handshake, signal marked $completion$ that indicates when the PLA has completed an evaluation or precharge operation. The $completion$ signal of a PLA switches high when the PLA completes an evaluation operation and switches low when it completes a precharge operation. Each PLA also has an internally generated clock signal (marked $INTCLK$). The PLA precharges when $INTCLK$ is low and evaluates when it is high.

The precharge operation of a PLA begins when P1 goes high, while evaluation starts when $P2$ rises, provided the *completion* signal of the PLA is low. After the *completion* signal of the topologically lowest level (level 1) PLAs goes low (PLA has precharged), the $P2$ signal of the topologically lowest level PLAs is asserted. This causes level 1 PLAs to evaluate. When the *completion* signal of the level 1 PLAs is asserted, the level 2 PLAs

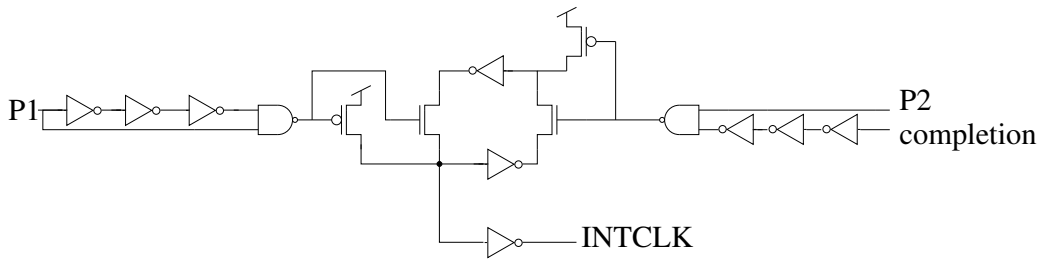


Fig. IX.2. Micropipelined PLA handshaking logic

begin evaluation. When the level 2 PLAs start evaluating (a short period after the INTCLK signal of the level 2 PLA rises), the level 1 PLAs start precharging. This ensures that the data from the PLAs of level 1 to the PLAs of level 2 is held until the PLAs of level 2 have latched the data from PLAs of level 1. This is necessary to make sure that data is not lost in the micro-pipeline. This handshaking mechanism is utilized across all PLA levels. Its implementation is shown in Figure IX.2.

The micropipelined structure in Figure IX.1 shows a single PLA at any topological level. In practice, there may be several PLAs at any level, in which case the *completion* signal for any level *i* would be generated by logically ANDing the *completion* signals of all PLAs of level *i*.

The screen capture of a Verilog simulation for a series of 4 PLAs showing the working of our handshaking protocol is shown in Figure IX.3. Note that this figure illustrates the asynchronous nature of the computation. In this figure, P2 is a signal from outside the micro-pipeline that signals the level 1 PLA to start evaluating (if the level 1 PLA is precharged). Once the level 1 PLA completes evaluation it signals the level 2 PLA to start evaluating. This happens at the time instant marked *a*, which occurs a short handshake period after the level 1 PLA completed its evaluation. We call this handshake period the *evaluation handshake period*. The level 2 PLA completes its evaluation at the time instant

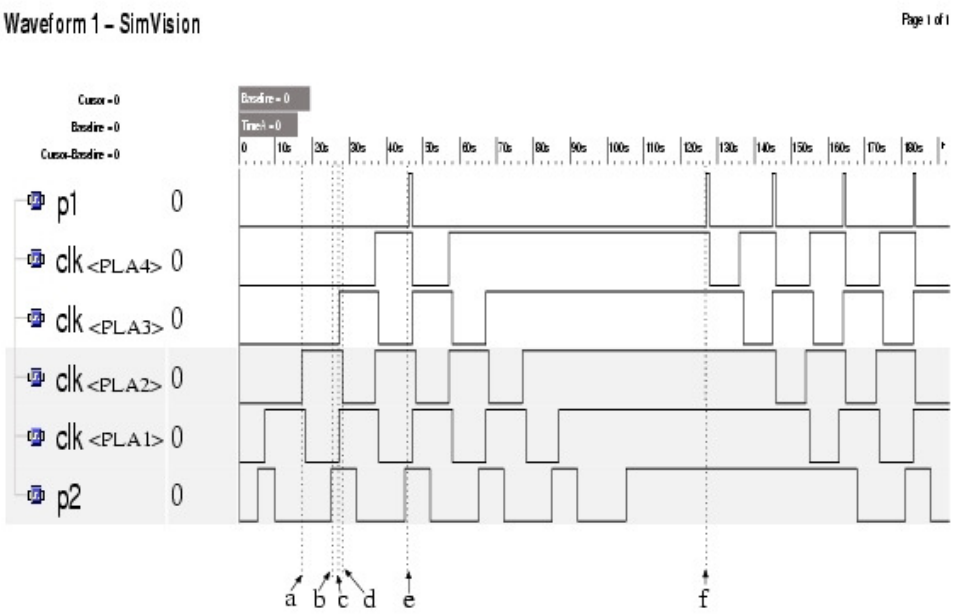


Fig. IX.3. Verilog simulation of our approach

marked *b* and then after a period equal to the evaluation handshake period, the level 3 PLA starts evaluating at the time instant marked *c*. A short-period after this (at the time instant marked *c*), the level 2 PLA starts precharging. We call this short-period the *precharge handshake period*. P1 is the user acknowledgement signal generated (at time instant marked *e*) after the PLA at level 4 completes its evaluation and the user has latched the data from the PLAs at this level. When the level 4 PLA receives this signal it starts precharging. If the user is late in acknowledging the data from the PLA at the last level, the pipeline is stalled till P1 is asserted again (at time instant marked *f*).

IX-B.2. Synthesis of Micropipelined PLA Networks

Synthesis of a PLA network for an asynchronous micropipelined implementation consists of a two step process. In the first step, we generate a NPLA from a multi-level logic netlist. In the second, we infer the stuttered signals that are induced by the synthesized result, and augment the netlist of the first part with stutter blocks which delay signals that traverse more than one level of PLAs.

In the first step, we begin by performing technology independent optimizations on the multi-level circuit C. Next, we decompose C into a network C^* of nodes with at most p inputs. In our experiments, $p = 5$. Now C^* is sorted in depth-first manner. The resulting array of nodes is sorted in *levelization*[1] order, and placed into an array L.

Now we greedily construct the logic in each PLA, by successively grouping nodes from L such that the resulting PLA implementation of the grouped nodes N^* does not violate the constraints of PLA width and height. This check is performed in a *check_PLA* routine, which first flattens N^* into a two-level form, P. It then calls *espresso* [76] on the result to minimize the number of cubes in P. Next, *check_PLA* calls a *PLA folding* routine which attempts to fold the inputs of P so as to implement a more complex PLA in the same area. Finally *check_PLA* ensures that the final PLA, after folding and simplification using *espresso*, satisfies the maximum width and height constraints respectively. If so, we attempt to include another node into N^*, otherwise we append the last PLA satisfying the height and width constraints to the result.

The *get_next_element* routine returns the most favorable node n among nodes in the fanout of nodes $n' \in N^*$ and nodes n'' which have the same level as the first node included into N^*, provided that the inclusion of n into N^* would not result in a cyclic PLA network. If

[1] Primary inputs are assigned a level 0, and other nodes are assigned a level which is one larger than the maximum level of all their fanins

Algorithm Decompose_Circuit_to_NPLA

C = optimize_network(C)

C^* = decompose_network(C, p)

L = dfs_and_levelize_nodes(C^*)

$N^* = 0$

$RESULT = 0$

while get_next_element(L) != NIL **do**

  $N^* = N^* \cup$ get_next_element(L)

  P = make_PLA(N^*)

  if check_PLA(P, W, H) **then**

    continue

  else

    Q = remove_last_element(N^*)

    $RESULT = RESULT \cup N^*$

    $N^* = Q$

  end if

end while

Fig. IX.4. Decomposition of a circuit into a network of PLAs

such nodes are not available, the first un-mapped node from L is returned. The favorability of a candidate is computed as:

$$favorability(n) = 2 \times [\#common\ fanins(n,n')] + [\#common\ fanouts(n,n')]$$

Nodes with shared fanins and fanouts decrease the number of PLAs created. We also found that shared fanins had a greater effect on this decrease. Hence in evaluating the favorability of a node we gave a greater weight to those nodes that shared a fanin with a node already included in the current PLA.

We implemented the algorithm to decompose a circuit into a network of PLAs in SIS [33]. The pseudo-code of the algorithm is shown in Figure IX.4.

The PLAs we used in our experiments had 16 inputs, 14 outputs and 24 rows. We found, through extensive experiments, that this size yielded a small number of PLAs and stutter blocks for a set of benchmark circuits.

Inferring of stuttered signals is performed by traversing the network of PLAs from inputs to outputs. For any output of a PLA of level l, if the PLAs in its fanout have a maximum level of l_j, then $l_j - l - 1$ stutter signals are inserted for this output, one for every level between l and l_j.

IX-B.3. Circuit Details of PLAs and Stutter Blocks

The PLA we use is a precharged NOR-NOR PLA (similar to the ones used in Chapters VII and VIII). The major difference between the PLAs utilized in this chapter and the ones utilized in Chapters VII and VIII is that the inputs have latches to store the data from a previous level. The schematic view of the PLA circuit is shown in Figure IX.5. The layout view of our PLA is shown in Figure IX.6. The operation of the PLA is as explained in Section VII-C (with *INTCLK* replacing *CLK*).

The INTCLK signal is generated from the *completion*, $P1$ and $P2$ signals using the circuit shown in Figure IX.2. On every rising edge of $P1$, a pulse is generated which

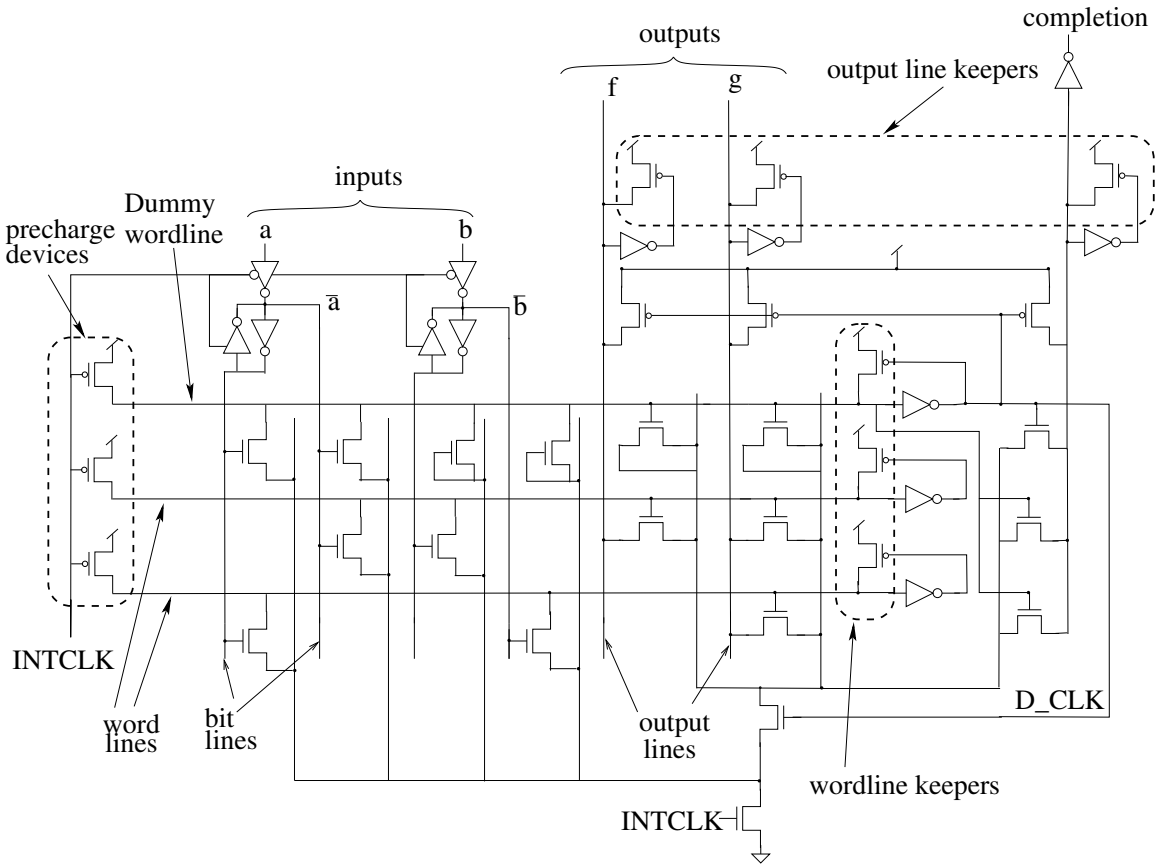


Fig. IX.5. Schematic view of the PLA

makes the INTCLK signal go low. forcing the PLA to enter the precharge phase. In other words, PLA *p* enters the precharge phase if PLAs at a level above the PLA *p* have started evaluation (after latching the input data). Once this happens, the *completion* signal of the PLA *p* falls (after all other signals in *p* have precharged). At this point, if *P*2 rises, then the PLA *p* enters the evaluation phase. In other words, if the PLA *p* has been precharged, and if the PLAs a level below complete their computation, then *p* enters the evaluation phase. The additional inverter(s) in the path of the *completion* signal are for design guard-banding. In our SPICE [32] simulation of this handshaking block, we found that it had a worst case delay of 25ns for INTCLK to fall, measured with respect to *P*1 rising. We called this the

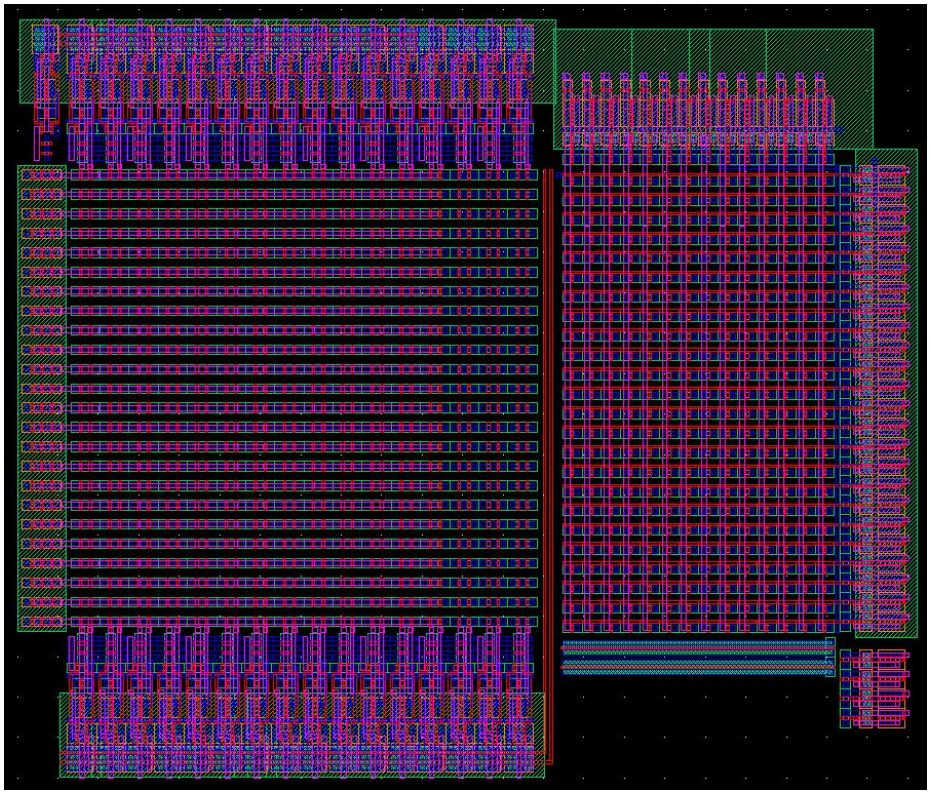


Fig. IX.6. Layout view of the PLA

precharge handshake period in Section IX-B.1. The handshaking block had a worst case delay of 60ns for INTCLK to rise (measured with respect to *completion* falling). We called this the *evaluation handshake period* in Section IX-B.1.

Note that each of the PLAs has a set of level sensitive latches on its inputs. When the PLA p has completed its computation, these latches hold their state, ensuring that the precharging of PLAs a level below does not change the state of the outputs of p that have been computed.

In this manner, odd levels of the NPLA precharge while even levels of PLAs evaluate.

The stutter block is simply a series of latches, implemented in the footprint of a PLA (in terms of height). Its function is to delay signals which traverse across levels of PLAs,

in order to guarantee correct operation under asynchronous micropipelining. For example, if there is a signal S_{jump1} that is an output of a level 1 PLA and is an input to a level 3 PLA, then a stutter block, consisting of a single latch, is placed between the two PLAs. The signal S_{jump1} is used as the data input to this latch and the data is latched using the INTCLK signal from level 2 PLA(s). This ensures that all the inputs to the level 3 PLA(s) are ready at the same time. For a signal traversing across n levels, n latches are required.

IX-C. Experimental Results

To compare the characteristics of an asynchronous micropipelined network of PLAs with that of a network of PLAs, we performed extensive simulations. All circuit simulations were done in SPICE [32], assuming a supply voltage of 0.2V and a temperature of $25^{c}irc$C and using 65nm BPTM [27] model cards. The area of the two design styles was computed using the sum of the areas of all the PLAs in the design, including the area of any stutter blocks (in the case of the micropipelined network of PLAs).

The asynchronous micropipelined network of PLAs has a throughput of

$T = \frac{1}{T_{eval}+T_{pchg}+2\cdot H_{eval}+H_{pchg}}$

Here T_{eval} is the evaluation delay of the PLA (recall, we utilize fixed sized PLAs in the design), T_{pchg} is the precharge delay of the PLA, H_{eval} is the evaluation handshake period and H_{pchg} is the precharge handshake period. The values of T_{eval}, T_{pchg}, H_{eval}, H_{pchg} are 210ns, 155ns, 60ns and 25ns respectively. As a consequence, the throughput is $\frac{1}{510ns}$. Note that the latency is still proportional to the number of PLA levels in the design, but the throughput is a constant.

In the traditional network of PLA implementation, all levels of PLAs are precharged together and then evaluate in a domino fashion. The timing diagram of this is shown in Figure VIII.6 in Chapter VIII. In case of the traditional network of PLA implementation,

the delay is given by the topological depth of the PLA network (in terms of number of PLAs) times the evaluation delay T_{eval} of each PLA. We also add to this the time taken to precharge all the PLAs in the design. Note that, in general, this is substantially greater than the throughput of our micro-pipelined approach.

We also compared the energy consumption of the two types of implementations. More specifically we compared the energy consumption per computation in the two types of NPLAs. For the micropipelined implementation, we first found (through SPICE simulation) the energy consumption for the operation of 1 PLA (over a period of 510ns) and multiplied this by the number of PLAs. To this, we add the energy consumption of the handshaking logic and the energy consumption in the stutter blocks. This gives us the energy consumption for one computation through the micropipelined NPLA.

While a micropipelined PLA spends very little time (equal to the handshaking periods) in a *precharged* state or *evaluated* state, the traditional NPLA spends substantial periods of time in the *precharged* state and *evaluated* state. This is evident from the timing diagram shown in Figure VIII.6. As a consequence, the micro-pipelined network of PLA based design wastes less energy in leakage than traditional network of PLA based designs.

Table IX.1. Comparison of micropipelined with traditional circuits

		# Stutter	Delay(ns) ↓			Energy(fJ) ↓			Area(μ^2) ↑		
Ckt	# PLAs	blocks	Non-μpipe	μpipe	Impr.	Non-μpipe	μpipe	Impr.	Non-μpipe	μpipe	Ovh
alu4	14	5	2885	510	5.66	5984.80	1811.43	3.30	9408	12768	1.36
apex6	24	12	2465	510	4.83	9033.09	3261.19	2.77	16128	24192	1.50
C432	11	4	2255	510	4.42	3877.22	1397.00	2.78	7392	10080	1.36
C499	14	4	2255	510	4.42	4961.02	1768.64	2.80	9408	12096	1.29
C880	16	5	2255	510	4.42	6088.11	2052.22	2.97	10752	14112	1.31
C1355	21	10	3305	510	6.48	10198.86	2863.68	3.56	14112	20832	1.48
C1908	24	13	3935	510	7.72	13814.19	3307.96	4.18	16128	24864	1.54
C2670	34	13	3515	510	6.89	18694.33	4472.11	4.18	22848	31584	1.38
C3540	67	46	7505	510	14.72	73900.56	9777.18	7.56	45024	75936	1.69
pair	65	35	4565	510	8.95	44442.77	9047.27	4.91	43680	67200	1.54
rot	19	13	3095	510	6.07	8966.68	2774.15	3.23	12768	21504	1.68
Avg	28.09	14.55			6.78			3.84			1.47

Table IX.1 reports the results of our experiments. The first column represents the cir-

cuit under study. The second column reports the number of PLAs required, while the third column reports the number of stutter blocks in the micro-pipelined network of PLAs. The next 3 columns report the delay of the non-micropipelined PLA, the throughput of the micropipelined PLA, and their ratio. Note that the throughput of the micropipelined PLAs is constant. The traditional PLA network delay is computed as described above. We note that the *micropipelined PLA results in a speedup of about 7× over a traditional design*. This is because in the micropipelined network of PLA circuit, the measure of delay is its throughput. Hence, for network of PLA circuits with larger topological depths, this improvement is more pronounced. Columns 7, 8 and 9 indicate that the energy consumption of the micropipelined NPLAs is about 4× lower than the energy consumption of the traditional NPLAs. The area penalty for the approach is about 47% on average, as indicated in the last three columns of Table IX.1.

IX-D. Optimum VDD for Micro-pipelined NPLAs

In the previous chapter (Chapter VIII), we discussed how the optimum supply voltage (VDD) that minimizes energy consumption for Network of PLAs depends on the logic depth of the network. The optimum VDD is higher for a circuit with a larger logic depth. This is due to the fact that while one PLA is precharging or evaluating, the other PLAs in the circuit waste energy in the idle *precharged* and *evaluated* states. In a micro-pipelined PLA, very little time is spent in these idle states. Hence the optimum VDD is expected to be low. The energy consumed by each PLA in a micro-pipelined Network of PLAs is equal to the sum of the energies spent in the *evaluating* and *precharging* states and the energies spent in the *precharged* states and *evaluated* states during the handshake periods. For our micro-pipeline, we hence estimate the energy consumed by each PLA to be given by the

following formula

$$Energy = PchgEnergy_{dyn} + EvalEnergy_{dyn} + \tag{9.1}$$
$$[PchgPwr_{sta} \times (H_{eval})] + [EvalPwr_{sta} \times (H_{eval} + H_{pchg})]$$

We characterized PLAs of different sizes to explore how the size of the PLA would affect the optimum VDD point. The results are given in Table IX.2. The PLAs were characterized using SPICE and the energy estimated using the Equation 9.1.

Table IX.2. Optimum VDD shift with PLA size

Size of PLA			Optimum VDD (in volts)			
# inputs	# outputs	# rows	at 25°C	at 50°C	at 75°C	at 100°C
16	14	24	0.22	0.28	0.30	0.30
16	10	16	0.22	0.28	0.30	0.30
12	6	12	0.20	0.28	0.28	0.30
8	4	8	0.18	0.22	0.22	0.28
4	2	4	0.15	0.18	0.20	0.22

As the data in Table IX.2 shows, the optimum VDD is low since the PLAs spend very little time in the *precharged* and *evaluated* states. However, we do notice that as the PLA get smaller, the optimum VDD does reduce. Also, just like we saw in the previous chapter, a higher temperature shifts the optimum VDD to higher value.

IX-E. Chapter Summary

In recent times, power consumption has become a dominant issue in VLSI circuit design. Sub-threshold circuit design is an appealing means to dramatically reduce this power consumption. However, sub-threshold designs suffer from the drawback of being significantly slower than traditional designs. In this chapter, we described a means to reclaim the speed penalty associated with sub-threshold designs. The approach is based on the use of a sub-threshold circuit design approach which is based on asynchronous micropipelining of a

levelized network of PLAs. We have developed a handshaking protocol, a circuit design approach and logic synthesis methodologies in this context. Our preliminary results demonstrate that by using our approach, a design can be sped up by $7\times$, with an area penalty of 47%. Further, the energy consumption of micropipelined NPLA based circuits is about $4\times$ lower than that of the traditional NPLAs circuits. Our simulations were validated in VERILOG, and circuit level characteristics were extracted using SPICE modeling. Using the techniques described in Chapter VIII, we also found that the optimal VDD for minimum energy operation of a micropipelined Network of PLAs can be above V_T (depending on the size of the PLA and the operating conditions). The techniques described in this chapter are equally applicable for these operating conditions as well.

CHAPTER X

CONCLUSIONS AND FUTURE DIRECTIONS

The focus of this dissertation is the problem of increasing leakage in modern VLSI designs. a study on leakage currents in VLSI design today. We have presented techniques to minimize as well as exploit leakage currents. In this chapter we summarize the work presented in this dissertation and present some avenues for further research.

In Chapter I of this dissertation we first motivated the need for low power design. We then pointed out why leakage power dominates dynamic power in VLSI design today. A description of the various sources of leakage was also presented.

Chapter II described some existing leakage reduction techniques. Three main classes of techniques were discussed - power gating, body biasing and input vector control. Each of these techniques have their pros and cons and there is no one "one-size-fits-all" technique that solves the leakage problem for all designs.

Chapter III described a new low-leakage standard cell based ASIC design methodology - the HL methodology. The philosophy of the HL technique is to ensure that during standby operation, the supply voltage is applied across more than one *off* device and there was at least one *off* high V_T device in the leakage path. This HL methodology requires the creation of two low leakage variants (H and L) of each standard cell in a library. By making sure that the core of the standard cells is not touched, we ensure that the effort involved in creating these variants is not too high, thus making the approach easy to adopt. The approach assumed that the primary inputs would be set to a pre-determined value in standby. The algorithm used in our approach to convert a regular standard-cell based design into a HL cell based design propagated these primary input values to first determine the state of the outputs of all gates in a design during standby, and then replaced them with their H or L variants. Experimental results proved that our HL methodology has better area and

delay characteristics than the popular MTCMOS technique. Also, unlike MTCMOS, the leakage in our methodology is precisely estimable, after an up-front characterization of the HL library. We also investigated the feasibility of using long-channel sleep transistors instead of high-V_T sleep transistors. We find that using high-V_T transistors in the HL cells (as opposed to using long-channel sleep transistors) gives a lower leakage with a similar delay penalty. However, if mask costs are a major constraint, then using long channel length sleep transistors may be more practical. In Chapter III we also discussed leakage reduction in domino logic.

As we move to newer process generations, the supply voltage is expected to scale down. The threshold voltages of both high-V_T and low-V_T devices is expected to scale down as well. To keep leakage low, the threshold voltages of high-V_T devices should be kept high. While this may make the delay of the HL approach worse, the delay gets worse for only one type of transition on each gate. In the traditional MTCMOS technique, both the rising and falling transitions would get worse. Therefore, the HL technique scales better than MTCMOS with newer process technologies.

A possible modification to the HL methodology could be the sharing of the header and footer sleep transistors. This would reduce the delay considerably. This sharing of transistors could help reduce the size of the sleep transistors too. However, the area impact of this is not clear. Such a sharing of sleep transistors would require the routing of the ungated power rails as well as the routing of the power rails gated by the (now shared) sleep transistors. One possible solution would be having the H variant cells and L variant cells placed in separate (alternate) rows of the standard-cell design. The sharing of sleep transistors also opens up a little-explored avenue of research - the sizing of the sleep transistors. Even in MTCMOS, when sleep transistors are shared, the sizing of these sleep transistors is a complex problem. The authors of [8] propose an MTCMOS sleep transistor sizing algorithm which is based on mutually exclusive discharging /charging of gates. While this

technique is easily applicable to regular circuits (like a chain of inverters or decoder logic), it is hard to utilize for random logic circuits. Similarly, a precise estimation of delay is also now dependent on knowing all the mutually exclusive discharge / charge patterns. There is room for research in the area of finding the worst case (largest delay) input pattern for MTCMOS circuits and circuit that use the HL methodology with shared sleep transistors.

Another area where improvements can be made in the HL methodology is in the technology mapping phase. In our implementation, the replacing of the regular cells with their H or L variants is dependent on the primary input vector. There are several heuristics (such as those in [15, 16, 18, 17, 20, 25, 24, 22, 21]) that can be used to find a minimal leakage primary input vector for the regular standard-cell based circuit. However, in our case once we find the best vector, we then modify the circuit (perform HL replacement). The solution we obtain is not necessarily the optimal solution, since it is quite likely that a different input vector that does not give the lowest leakage in the regular standard-cell based circuit, gives a lower leakage in the HL-cell based circuit.

We have noticed that the HL approach worsens delay, but only for one transition for the gate. This fact can be exploited through another possible extension to the HL methodology is replacing the regular standard-cells with HL cells such that the critical delay is bounded. This would involve first finding all the critical paths in a design. If a critical path utilizes the pullup network of a gate, then we would attempt to replace that gate with a H variant. Similarly we would attempt to replace a gate with an L variant if the pulldown network of the gate is in the critical path. Yet another possible extension to the HL methodology is to create the technology mapping library so that it contains both the regular standard-cells as well as their HL counterparts. We could then perform technology mapping with leakage added as one of the objectives of the mapper. The resulting circuit would contain a mix of regular standard-cells and HL cells, with the HL cells used in the off-critical paths.

While most leakage reduction approaches (such as the HL and MTCMOS approaches)

have a delay penalty, in Chapter IV, we presented an approach that reduces leakage while ensuring that there was no delay penalty (and in many cases a small delay improvement). We proposed an approach which combined circuit modification and input vector control at a fine-grained level. Our approach involved traversing a given circuit topologically from inputs to outputs, selectively modifying a gate so that its output (in sleep mode) is in a state which helps minimize the leakage of other gates in its transitive fanout. For this modification we developed different variants of each cell in a library, including some cells that allowed an output to be 'split'. While traditional input vector control only allows the primary input vector to be set so as to minimize leakage, our approach focused on circuit modifications that allowed us to not only set primary input values to a known state, but also control the logic values of internal nodes (in the standby/sleep mode). One of the key advantages of our technique is that we are able to achieve a leakage of about 30% (over input vector control alone) without a delay penalty. While other techniques such as HL or MTCMOS can get achieve greater leakage savings, these techniques are orthogonal to our approach and these techniques have an associated delay penalty. Also, these approaches involve additional mask costs to create the high-V_T transistors. The approach presented in Chapter IV does not use multiple V_T transistors and is hence less expensive to implement.

Our algorithm currently replaces gates in a circuit to allow control of internal node signals (while ensuring that critical delay is not increased). If we allowed the algorithm to perform resizing of the sleep cut-off transistors used in the variants of the standard-cells, we could potentially use the available slack better and achieve further leakage reductions. Sharing of the sleep cut-off transistors used is another possible improvement to the methodology. The algorithm implemented currently is a simple one that traverses a given circuit from input to output. While this makes the algorithm fast, the solution we get may not be optimal. One possible modification to our algorithm would be to first find the lowest leakage input vector, propagate this through the circuit and then target high leakage gates

and try to control their inputs.

In Chapter V, we first present results (from a 130nm test-chip) that prove that while reverse body biasing (RBB) reduces sub-threshold leakage, the BTBT leakage component increases with greater applied RBB. Hence, there is an optimum RBB point. We presented a scheme that monitors the leakage through a representative device and finds this optimum RBB point. The scheme consists of a leakage current monitor (LCM), a programmable body bias voltage generator and digital block to interface with the LCM and the body bias voltage generator. The LCM worked by essentially converting the problem of measuring the leakage current into one of measuring the time taken for a representative leaking device to discharge (in the case of a leaking NMOS device) or charge (in the case of a leaking PMOS device) a capacitively loaded node. To cope with the large range in leakage currents, the LCM used a tunable bank of capacitors and an adjustable gate bias. The scheme presented incurred a very reasonable placed-and-routed area and also had a very small power consumption. Since the LCM presented in this chapter is small in area and not power-hungry, it could be distributed on different portions of an IC and used to monitor the leakage currents at these different points. This could be potentially useful to a designer or researcher investigating intra-die leakage variations.

The leakage reduction techniques presented in Chapters III, IV and V are all techniques easily applicable to traditional IC design today. The techniques presented in Chapters III and IV involve some initial work in modifying or augmenting the standard-cell library. However, this task is done exactly once, upfront. There are several companies in the semiconductor industry that build standard-cell libraries. Some of them already offer low leakage standard-cell variants as part of their libraries. The variants presented in Chapters III and IV, along with the design flow and methodology to use them, could potentially be offered by these companies as part of their low leakage standard-cell libraries. Some companies also sell blocks of logic and circuitry as Intellectual Property (IP) cores. The

scheme presented in Chapter V is one that has potential to be offered as one such IP core.

While Chapters II, III, IV and V discussed leakage reduction techniques, the remaining chapters of this dissertation after (and including) Chapter VI all focus on leakage *exploitation*. In Chapter VI we first presented data from some exploratory studies that revealed the opportunity that sub-threshold circuit design offers. The main advantages of sub-threshold circuits are:

- Low power consumption and heat dissipation
- Smaller delays with increasing temperature
- High PDP (Power Delay Product)

We also presented the three main disadvantages facing sub-threshold circuit design today:

- Large delay
- Sensitivity to process, voltage and temperature (PVT) variations
- Lack of a systematic EDA framework to implement sub-threshold circuits.

This chapter also discussed the application space for sub-threshold design. The remaining chapters of this dissertation proposed techniques to address each of the disadvantages cited above.

In Chapter VII we presented a way to make a sub-threshold circuit less sensitive to PVT variations. We proposed a sub-threshold design approach which dynamically compensates for inter and intra-die PVT variations. The approach we proposed involved adaptively adjusting the body bias to dynamically stabilize the delay of the circuit. In the proposed approach a multi-level network of medium sized Programmable Logic Arrays (PLAs) was the circuit implementation structure. The approach used a global *beat clock* and attempted to "phase lock" the delay of a representative PLA (in a cluster of localized PLAs) to the

beat clock. This phase locking was done in a closed-loop fashion using a phase-detector and charge-pump which charged or discharged the bulk node of the NMOS devices in the PLAs. The PLAs we used were dynamic (NOR-NOR) PLAs. In such PLAs, the critical delay (the evaluation delay) is dependent mainly on the NMOS devices in the core of the PLA. Hence, we only controlled the bulk nodes of the NMOS devices. Simulation results (using 65nm BSIM4 model cards from [27]) proved that our adaptive body biasing scheme is very effective. An analysis of the loop gain of the closed-loop adaptive body biasing scheme was also presented. We found that the width of the charge-pump transistors and the capacitance of the bulk node can be used to tune the response of the scheme. Sub-threshold circuits are extremely sensitive to PVT variations. A compensating scheme such as the one presented in Chapter VII is crucial for any practical sub-threshold design.

While a lower voltage reduces power consumption it also worsens the time taken to perform a computation. As result the energy consumed in performing a computation can actually be a higher for a circuit utilizing a lower operating voltage. The optimum voltage for minimum energy is in fact dependent on the circuit topology. In Chapter VIII, we studied the problem of finding the optimum voltage for minimum energy in the context of designing a circuit using a network of dynamic NOR-NOR PLAs. We derived a method to calculate the energy consumed by a network of medium (fixed) sized PLAs by just characterizing *one* of the PLAs in the network. Using this method we estimated the energy for networks of PLAs of various logic depths. We found that as the logic depth of a circuit got larger, the optimum VDD became higher. This is because when one PLA in a network is evaluating or precharging, the other PLAs in the network (at a different logic depth) are in the evaluated or precharged idle states, wasting leakage power. The dependence of the optimum VDD on circuit topology holds for other circuit design styles as well, not just for a network of PLA based design.

In Chapter IX we proposed using asynchronous micropipelining to help improve the

throughput of sub-threshold circuits and hence reduce the speed gap between sub-threshold and traditional circuits. The approach used a network of PLA based design flow similar to the flow used in Chapters VII and VIII. The synthesis algorithm used in the design flow was augmented to allow the network of PLAs to be micro-pipelined. On a set of benchmark circuits, the micropipelined approach was found to give a 7× improvement in throughput over a non-micropipelined network of PLAs. After applying the micro-pipelining approach, the delay of a sub-threshold circuit is approximately 1.5-4× worse than a traditional super-threshold circuit. Without this technique, recall the delay penalty was 10-25×. The micro-pipelined circuits were also found to be more energy efficient due to the fact that little time and energy was wasted in the idle precharged and evaluated states. Using the concepts of Chapter VIII, we studied how the optimum VDD for an asynchronous micro-pipelined circuit would change with PLA size and temperature. We found that in a majority of cases, the optimum VDD for minimum energy was slightly above the threshold voltage of the NMOS devices. The micro-pipelining technique is applicable in these near-threshold regions of operation as well.

In Chapters VII, VIII and IX we proposed using a network of PLAs to design sub-threshold circuits. In Chapters VII and IX we presented approaches to respectively tackle the issues of sensitivity of sub-threshold circuits to PVT variations, and the problem of increased delay of sub-threshold circuits. We also proposed design flows to implement digital circuits as a sub-threshold network of PLAs. As discussed in [63], using a network of medium-sized PLAs is a suitable way to implement structured ASICs with a low NRE. Structured ASICs allow designs to be implemented using very few lithography masks (metal and via masks only in the case of [63]). The sub-threshold design approaches presented here are, hence, very easily applied to a structured ASIC setting as well.

In a sub-threshold design, a high-quality power and ground distribution network is crucial since the operating voltages are extremely low. Also, such a circuit can be suscep-

tible to noise. In such a scenario, a layout fabric [65, 66] is ideally suited for sub-threshold circuits. The network of PLAs used in our sub-threshold circuit deign flows is naturally amenable to such a fabric. One of the reasons for the success of traditional standard-cell based CMOS design technology is the existence of a design flow and methodology that made the design of standard-cell based ICs practical and feasible. The sub-threshold design approaches in this dissertation are presented to provide a design flow and methodology that can help make sub-threshold circuit design practical and feasible.

Sub-threshold circuits are useful in applications where minimum power and energy consumption is most important while performance is a secondary requirement. Examples of such applications are sensor networks, digital wrist watches and medical equipment such as hearing aids. Another possible application for sub-threshold circuits is the following - in the near future, we could have devices implanted within our bodies, which monitor the status of our health. These devices could probably derive their energy from the heat in the body or the flow of blood. These devices will be required to consume and dissipate extremely low amounts of power not only because the energy available is limited, but also because the heat dissipated by the device should not affect the surrounding tissue that it is implanted in. In such applications, sub-threshold designs are probably going to be the only feasible choice. With a large market for such low power devices, sub-threshold circuit design could become as popular as traditional CMOS design. The sub-threshold design approaches presented in this dissertation should help accelerate the adoption of such devices.

REFERENCES

[1] "Microprocessor Power Consumption," www.intel.com, accessed on 5th May, 2005.

[2] K Roy, S Mukhopadhyay, and H Mahmoodi-Meimand, "Leakage Current Mechanisms and Leakage Reduction Techniques in Deep-Submicrometer CMOS Circuits," *Proc. IEEE*, vol. 91, no. 2, pp. 305–327, Feb 2003.

[3] R L Aguiav and D M Santos, "Modelling Charge-pump Delay Locked Loops," in *Proc. International Conference on Electronics, Circuits and Systems*, Pafos, Cyprus, Sep 1999, pp. 823–826.

[4] "The International Technology Roadmap for Semiconductors," http://public.itrs.net/, 2003, accessed on 12th Nov, 2003.

[5] W Daasch, C Lim, and G Cai, "Design of VLSI CMOS Circuits under Thermal Constraint," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 49, no. 8, pp. 589–593, Aug 2002.

[6] N Weste and K Eshraghian, *Principles of CMOS VLSI Design - A Systems Perspective*, Addison-Wesley, Reading, MA, 1988.

[7] J Rabaey, *Digital Integrated Circuits: A Design Perspective*, Prentice Hall, Upper Saddle River, NJ.

[8] J T Kao and A P Chandrakasan, "Dual-threshold Voltage Techniques for Low-power Digital Circuits," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 7, pp. 1009–1018, Jul 2000.

[9] S Mutoh, T Douseki, Y Matsuya, T Aoki, S Shigematsu, and J Yamada, "1-V Power Supply High-speed Digital Circuit Technology with Multithreshold-voltage CMOS," *IEEE Journal of Solid-State Circuits*, vol. 30, no. 8, pp. 847–854, Aug 1995.

[10] K Kumagai, H Iwaki, H Yoshida, H Suzuki, T Yamada, and S Kurosawa, "A Novel Powering-down Scheme for Low Vt CMOS Circuits," in *Digest of Technical Papers, Symposium on VLSI Circuits*, Honolulu, HI, Jun 1998, pp. 44–45.

[11] T Kuroda, T Fujita, S Mita, T Nagamatsu, S Yoshioka, K Suzuki, F Sano, M Norishima, M Murota, M Kako, M Kinugawa M Kakumu, and T Sakurai, "A 0.9-V, 150-MHz, 10-mW, 4 *mm* 2, 2-D Discrete Cosine Transform Core Processor with Variable Threshold-voltage (VT) Scheme," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 11, pp. 1770–1779, Nov 1996.

[12] T Inukai, T Hiramoto, and T Sakurai, "Variable Threshold Voltage CMOS (VTCMOS) in Series Connected Circuits," in *Proc. International Symposium on Low Power Electronics and Design*, Huntington Beach, CA, 2001, pp. 201–206.

[13] Im Hyunsik, T Inukai, H Gomyo, T Hiramoto, and T Sakurai, "VTCMOS Characteristics and Its Optimum Conditions Predicted by a Compact Analytical Model," in *Proc. International Symposium on Low Power Electronics and Design*, Huntington Beach, CA, 2001, pp. 123–128.

[14] F Assaderaghi, D Sinitsky, S A Parke, J Bokor, P K Ko, and C Hu, "Dynamic Threshold-voltage MOSFET (DTMOS) for Ultra-low Voltage VLSI," *IEEE Transactions on Electron Devices*, vol. 44, no. 3, pp. 414–422, Mar 1997.

[15] J Halter and F Najm, "A Gate-Level Leakage Power Reduction Method for Ultra Low Power CMOS Circuits," in *Proc. Custom Integrated Circuits Conference*, Santa Clara, CA, 1997, pp. 475–478.

[16] C Zhanping, M Johnson, W Liqiong, and W Roy, "Estimation of Standby Leakage Power in CMOS Circuit Considering Accurate Modeling of Transistor Stacks," in *Proc. International Symposium on Low Power Electronics and Design*, Monterey, CA, 1998, pp. 239–244.

[17] M Johnson, D Somasekhar, and K Roy, "Models and Algorithms for Bounds on Leakage in CMOS Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 18, no. 6, pp. 714–725, Jun 1999.

[18] F Gao and J Hayes, "Exact and Heuristic Approaches to Input Vector Control for Leakage Power Reduction," in *Proc. International Conference on Computer-aided Design*, San Jose, CA, Nov 2004, pp. 527–532.

[19] K Chopra and S Vrudhula, "Implicit Pseudo Boolean Enumeration Algorithms for Input Vector Control," in *Proc. Design Automation Conference*, San Diego, CA, Jun 2004, pp. 767–772.

[20] K Gulati, N Jayakumar, and S Khatri, "An Algebraic Decision Diagram (ADD) Based Technique to Find Leakage Histograms of Combinational Designs," in *Proc. International Symposium on Low Power Electronic Design*, San Diego, CA, Aug 2005.

[21] R. I. Bahar, E. A. Frohm, C. M. Gaona, G. D. Hachtel, E. Macii, A. Pardo, and F. Somenzi, "Algebraic Decision Diagrams and Their Applications," *Formal Methods in Systems Design*, vol. 10, no. 2/3, pp. 171–206, 1997.

[22] R Rao, F Liu, J Burns, and R Brown, "A Heuristic to Determine Low Leakage Sleep State Vectors for CMOS Combinational Circuits," in *Proc. International Conference on Computer-aided Design*, San Jose, CA, Nov 2003, pp. 689–692.

[23] M. Abramovici, M. A. Breuer, and A. D. Friedman, *Digital Systems Testing and Testable Design*, IEEE Press, New York, NY, 1990.

[24] F Aloul, S Hassoun, K Sakallah, and D Blauuw, "Robust SAT-based Search Algorithm for Leakage Power Reduction," in *Proc. Power and Timing Models and Simulation*, Seville, Spain, 2002.

[25] K Gulati, N Jayakumar, and S P Khatri, "A Probabilistic Method to Determine the Minimum Leakage Vector for Combinational Designs," in *Proc. International Symposium on Circuits and Systems.*

[26] "BSIM3 Homepage," http://www-device.eecs.berkeley.edu/∼bsim3/intro.html, accessed on 5th June 2004.

[27] Y Cao, T Sato, D Sylvester, M Orshansky, and C Hu, "New Paradigm of Predictive MOSFET and Interconnect Modeling for Early Circuit Design," in *Proc. IEEE Custom Integrated Circuit Conference*, Orlando, FL, Jun 2000, pp. 201–204, http://www-device.eecs.berkeley.edu/ ptm.

[28] M Horiguchi, T Sakata, and K Itoh, "Switched-Source-Impedance CMOS Circuit for Low Standby Subthreshold Current Giga-scale LSI's," *IEEE Journal of Solid-State Circuits*, vol. 28, no. 11, pp. 1131–1135, Nov 1993.

[29] D Takashima, S Watanabe, H Nakano, Y Oowaki, K Ohuchi, and H Tango, "Standby/Active Mode Logic for Sub-1-V Operating ULSI Memory," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 4, pp. 441–447, Apr 1994.

[30] K-S Min, H Kawaguchi, and T Sakurai, "Zigzag Super Cut-off CMOS (ZSCCMOS) Block Activation with Self-adaptive Voltage Level Controller: An Alternative to Clock-gating Scheme in Leakage Dominant Era," in *Digest of Technical Papers,*

International Solid-State Circuits Conference, San Francisco, CA, 2003, vol. 1, pp. 400–502.

[31] T Burd, *CMOS Standard Cell 2_3lp Library Documentation*, University of California, Berkeley, Mar 1994.

[32] L Nagel, "SPICE: A Computer Program to Simulate Computer Circuits," in *University of California, Berkeley UCB/ERL Memo M520*, May 1995.

[33] E M Sentovich, K J Singh, L Lavagno, C Moon, R Murgai, A Saldanha, H Savoj, P R Stephan, R K Brayton, and A L Sangiovanni-Vincentelli, "SIS: A System for Sequential Circuit Synthesis," Tech. Rep. UCB/ERL M92/41, University of California, Berkeley, CA 94720, May 1992.

[34] P C McGeer, A Saldanha, R K Brayton, and A L Sangiovanni-Vincetelli, *Delay Models and Exact Timing Analysis*, chapter 8, Logic Synthesis and Optimization. Kluwer Academic Publishers, New York, NY, 1993.

[35] Cadence Design Systems, Inc., 555 River Oaks Parkway, San Jose, CA 95134, USA, *Envisia Silicon Ensemble Place-and-route Reference*, Nov 1999.

[36] P Gupta, A B Kahng, P Sharma, and D Sylvester, "Gate-length Biasing for Runtime-leakage Control," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 8, pp. 1475–1485, Aug 2006.

[37] A B Kahng, S Muddu, and P Sharma, "Impact of Gate-length Biasing on Threshold-voltage Selection," in *Proc. International Symposium on Quality Electronic Design*, Santa Clara, CA, Mar 2006, pp. 27–29.

[38] V Kursun and E G Friedman, "Low Swing Dual Threshold Voltage Domino Logic," in *Proc. IEEE Great Lakes Symposium on VLSI*, New York, NY, April 2002, pp.

47–52.

[39] A Abdollahi, F Fallah, and P Massoud, "Runtime Mechanisms for Leakage Current Reduction in CMOS VLSI Circuits," in *Proc. 2002 International Symposium on Low Power Electronics and Design*, Monterey, CA, 2002, pp. 213–218.

[40] A Abdollahi, F Fallah, and M Pedram, "Leakage Current Reduction in CMOS VLSI Circuits by Input Vector Control," *IEEE Transactions on VLSI Systems*, vol. 12, no. 2, pp. 140–154, 2004.

[41] Lin Yuan and Gang Qu, "Enhanced Leakage Reduction Technique by Gate Replacement," in *Proc. Design Automation Conference*, 2005, pp. 47–50.

[42] L Cheng, L Deng, D Chen, and M D F Wong, "A Fast Simultaneous Input Vector Generation and Gate Replacement Algorithm for Leakage Power Reduction," in *Proc. Design Automation Conference*, San Francisco, CA, 2006, pp. 117–120.

[43] A Keshavarzi, S Narendra, S Borkar, C Hawkins, K Royi, and V De, "Technology Scaling Behavior of Optimum Reverse Body Bias for Standby Leakage Power Reduction in CMOS ICs," in *Proc. International Symposium on Low Power Electronics and Design*, San Diego, CA, Aug 1999, pp. 252–254.

[44] C Neau, "Personal communication," Purdue University, West Lafayette, Indiana, Jun 2004.

[45] C Neau and K Roy, "Optimal Body Bias Selection for Leakage Improvement and Process Compensation over Different Technology Generations," in *Proc. International Symposium on Low Power Electronics and Design*, Seoul, Korea, Aug 2003, pp. 116 – 121.

[46] X Liu and S Mourad, "Performance of Submicron CMOS Devices and Gates with Substrate Biasing," in *The IEEE International Symposium on Circuits and Systems*, Geneva, Switzerland, May 2000, vol. 4, pp. 9–12.

[47] Y-S Lin, C-C Wu, C-S Chang, R-P Yang, W-M Chen, J-J Liaw, and C Diaz, "Leakage Scaling in Deep Submicron CMOS for SoC," *IEEE Transactions on Electron Devices*, vol. 49, no. 6, pp. 1034–1041, Jun 2002.

[48] S Mukhopadhyay, H Mahmoodi-Meimand, C Neau, and K Roy, "Leakage in Nanometer Scale CMOS Circuits," in *Proc. International Symposium on VLSI Technology, Systems, and Applications*, Hsinchu, Taiwan, 2003, pp. 307–312.

[49] J Chen, S Wong, and Y Wang, "An Analytic Three-terminal Band-to-band Tunneling Model on GIDL in MOSFET," *IEEE Transactions on Electron Devices*, vol. 48, no. 7, pp. 1400–1405, Jul 2001.

[50] Y Taur and T H Ning, *Fundamentals of Modern VLSI Devices*, Cambridge University Press, New York, NY, 1998.

[51] T Kobayashi and T Sakurai, "Self-adjusting Threshold-voltage Scheme (SATS) for Low-voltage High-speed Operation," in *Proc. IEEE Custom Integrated Circuits Conference*, San Diego, CA, May 1994, pp. 271–274.

[52] H Soeleman, K Roy, and B Paul, "Robust Subthreshold Logic for Ultra-low Power Operation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 9, no. 1, pp. 90–99, 2001.

[53] S-H Choi, B-K Kim, J Park, C-H Kang, and D-S Eom, "An Implementation of Wireless Sensor Network," *IEEE Transactions on Consumer Electronics*, vol. 50, no. 1, pp. 236–244, Feb 2004.

[54] "The MultimodAl NeTworks of In-situ Sensors (MANTIS) project," http://mantis.cs.colorado.edu, 2004.

[55] A Abidi, G Pottie, and W Kaiser, "Power-conscious Design of Wireless Circuits and Systems," *in Proceedings of the IEEE*, vol. 88, no. 10, pp. 1528–1545, Oct 2000.

[56] M Mui, K Banerjee, and A Mehrotra, "Power Supply Optimization in Sub-130 nm Leakage Dominant Technologies," in *Proc. 5th International Symposium on Quality Electronic Design*, San Jose, CA, Mar 2004, pp. 409–414.

[57] K Kanda, K Nose, K Kawaguchi, and T Sakurai, "Design Impact of Positive Temperature Dependence on Drain Current in sub-1-V CMOS VLSIs," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 10, pp. 1559–1564, Oct 2001.

[58] H Soeleman, K Roy, and B Paul, "Robust Subthreshold Logic for Ultra-low Power Operation," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 9, no. 1, pp. 90–99, Feb 2001.

[59] H Soeleman and K Roy, "Digital CMOS Logic Operation in the Sub-threshold Region," in *Proc. Tenth Great Lakes Symposium on VLSI*, Chicago, IL, Mar 2000, pp. 107–112.

[60] H Soeleman and K Roy, "Ultra-low Power Digital Subthreshold Logic Circuits," in *Proc. International Symposium on Low Power Electronic Design*, San Diego, CA, 1999, pp. 94–96.

[61] B Paul, H Soeleman, and K Roy, "An 8X8 Sub-Threshold Digital CMOS Carry Save Array Multiplier," in *Proc. European Solid State Circuits Conference*, Villach, Austria, Sept 2001, pp. 377–380.

[62] J Tschanz, J Kao, S Narendra, R Nair, D Antoniadis, A Chandrakasan, and V De, "Adaptive Body Bias for Reducing Impacts of Die-to-Die and Within-die Parameter Variations on Microprocessor Frequency and Leakage," vol. 37, pp. 1396–1402, Nov 2002.

[63] N Jayakumar and S Khatri, "A METAL and VIA Maskset Programmable VLSI Design Methodology Using PLAs," in *Proc. IEEE/ACM International Conference on Computer Aided Design*, San Jose, CA, Nov 2004, pp. 590–594.

[64] P Zarkesh-Ha, T Mule, and J D Meindl, "Characterization and Modelling of Clock Skew with Process Variation," in *Proc. IEEE Custom Integrated Circuits Conference*, San Diego, CA, May 1999, pp. 441–444.

[65] S Khatri, A Mehrotra, R Brayton, A Sangiovanni-Vincentelli, and R Otten, "A Novel VLSI Layout Fabric for Deep Sub-Micron Applications," in *Proc. Design Automation Conference*, New Orleans, LA, Jun 1999.

[66] S P Khatri, R K Brayton, and A Sangiovanni-Vincentelli, "Cross-talk Immune VLSI Design Using a Network of PLAs Embedded in a Regular Layout Fabric," in *Proc. IEEE/ACM International Conference on Computer Aided Design*, San Jose, CA, Nov 2000, pp. 412–418.

[67] J Rabaey, "Design at the End of the Silicon Roadmap," Keynote Talk, Asia and South Pacific Design Automation Conference, Jan 2005.

[68] R Gonzalez, B M Gordon, and M A Horowitz, "Supply and Threshold Voltage Scaling for Low Power CMOS," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 8, pp. 1210–1216, Aug 1997.

[69] A Wang, A Chandrakasan, and S Kosonocky, "Optimal Supply and Threshold Scaling

for Subthreshold CMOS Circuits," in *Proc. IEEE Computer Society Annual Symposium on VLSI*, Apr 2003, pp. 5–9.

[70] B H Calhoun, A Wang, A Chandrakasan, and S Kosonocky, "Device Sizing for Minimum Energy Operation in Subthreshold Circuits," in *Proc. IEEE Custom Integrated Circuits Conference*, Orlando, FL, Oct 2004, pp. 95–98.

[71] B Zhai, D Blaauw, D Sylvester, and K Flautner, "Theoretical and Practical Limits of Dynamic Voltage Scaling," in *Proc. Design Automation Conference*, San Diego, CA, Jun 2004, pp. 868–873.

[72] F Mo and R Brayton, "PLA-based Regular Structures and their Synthesis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 6, pp. 723–729, Jun 2003.

[73] F Mo and R Brayton, "River PLAs: A Regular Circuit Structure," in *Proc. Design Automation Conference*, New Orleans, LA, Jun 2002, pp. 201–206.

[74] F Mo and R Brayton, "Whirlpool PLAs: A Regular Logic Structure and Their Synthesis," in *Proc. IEEE/ACM International Conference on Computer Aided Design*, San Jose, CA, Nov 2002, pp. 543–550.

[75] S P Khatri, "Cross-talk Noise Immune VLSI Design using Regular Layout Fabrics," Ph.D. dissertation, University of California, Berkeley, Dec 1999.

[76] R K Brayton, G D Hachtel, C T McMullen, and A Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*, Kluwer Academic Publishers, New York, NY, 1984.

VITA

Nikhil Jayakumar received his Bachelor's degree in electrical and electronics engineering from the University of Madras, India in 2001 and his Masters degree in electrical engineering from the University of Colorado at Boulder in 2003. He received a Doctoral degree in computer engineering from the Department of Electrical & Computer Engineering at Texas A&M University, College Station. During his graduate and doctoral studies he did research and published papers in many aspects of VLSI including formal verification, clock network design, routing, structured ASIC design, radiation-hard design, logic synthesis, LDPC decoder architectures, statistical timing and low power design. His current research focus is on low power design and more specifically on techniques to reduce as well as exploit leakage currents in VLSI.

Nikhil Jayakumar may be reached at the Department of Electrical and Computer Engineering 333 WERC, Texas A&M University, College Station, TX 77843-3259. His email address is: nikhil_AT_ece_DOT_tamu_DOT_edu.