Process-Variation-Resistant Dynamic Power Optimization for VLSI Circuits

Except where reference is made to the work of others, the work described in this dissertation is my own or was done in collaboration with my advisory committee. This dissertation does not include proprietary or classified information.

Fei Hu

Certificate of Approval:

Foster Dai
Associate Professor
Electrical and Computer Engineering

Vishwani D. Agrawal, Chair
James J. Danaher Professor
Electrical and Computer Engineering

Darrel Hankerson
Professor
Mathematics and Statistics

Stephen L. McFarland
Acting Dean
Graduate School

Process-Variation-Resistant Dynamic Power Optimization for VLSI Circuits

Fei Hu

A Dissertation

Submitted to

the Graduate Faculty of

Auburn University

in Partial Fulfillment of the

Requirements for the

Degree of

Doctor of Philosophy

Auburn, Alabama
May 11, 2006

Process-Variation-Resistant Dynamic Power Optimization for VLSI Circuits

Fei Hu

_____

Signature of Author

_____

Date of Graduation

Vita

Fei Hu was born on October 27, 1975 in Yueyang, Hunan Province, P.R.China. He entered the University of Electronic Science and Technology of China (UESTC) in 1992 and received a B.S. in Image Processing and Transmission in 1996. He received his second B.S. in Industrial Foreign Trade from UESTC in 1998. In August 2000, he started his graduate study at Auburn University and obtained an M.S. in Electrical and Computer Engineering in December 2002. He has been a student member of the Institute of Electrical and Electronics Engineers since 2001.

Dissertation Abstract

Process-Variation-Resistant Dynamic Power Optimization for VLSI Circuits

Fei Hu

Doctor of Philosophy, May 11, 2006
(M.S., Auburn University, 2002)
(2nd B.S., University of Electronic Science and Technology of China, 1998)
(B.S., University of Electronic Science and Technology of China, 1996)

197 Typed Pages

Directed by Vishwani D. Agrawal

Power dissipation is an increasingly critical issue in modern VLSI design and testing. Previously, linear programming (LP) based methods have been proposed for optimization of circuits for low power dissipation. However, as the transistor size shrinks, variations in the device and circuit parameters increase. Under the existence of process-variations, a circuit optimized by previous techniques will not be able to maintain the low power dissipation.

In this dissertation, we investigate dynamic power optimization techniques that are resistant to the process variation. That is, the power dissipation of the optimized circuit should maintain low power dissipation even if certain degree of process-variation exists. We consider process-variation in terms of the delay variations and classify them into the inter-die and intra-die variations. We prove that the inter-die variation has negligible effect on the power dissipation of the circuit.

We propose two new linear programming (LP) models to obtain solutions that continue to maintain low power dissipation under the process variation. The two LP models are based on worst-case timing analysis and statistical timing analysis, respectively. We also consider

input-vector specific optimization to reduce the number of delay elements inserted into the circuit. Our experimental results show that our LP models can obtain a more process-variation-resistant solution in terms of both power dissipation and critical delay. That is, our optimization is also able to suppress the deviation of critical delay from its nominal value under the process-variation. We use a trade-off between the robustness (process-variation-resistance) and the circuit performance in terms of the critical delay. Our experimental results on ISCAS'85 benchmarks show complete suppression of power variation for small circuits and process-variations. Up to 53% reduction of power variation and 40% reduction of the delay variation are obtained for those large circuits with a large process-variation. In our experiments, the application of input-specific optimization to our LP model of Chapter 5 is able to reduce the number of buffers by up to 63%.

Our work explores a new aspect of generalized dynamic power optimization techniques. We propose a LP based method to improve a design under the existence of process-variation. The resulting circuit is more process-variation-resistant in terms of both power dissipation and critical delay. The merit of our solution will be increasingly vital as technology keeps marching forward.

Acknowledgments

First, I would like to thank my advisor Dr. Vishwani D. Agrawal for all his generous supports, the academic freedom provided in researches, and many helpful suggestions throughout my Ph.D. study. It has been my great pleasure and honor to work with Dr. Agrawal towards my Ph.D. degree.

I would also like to thank other committee members, Dr. Foster Dai and Dr. Darrel Hankerson for their sincere help and effort. Thanks are due to other faculty in the VLSI Design and Testing group, Dr. Adit Singh, Dr. Charles Stroud and Dr. Victor Nelson, for all the helpful discussions and suggestions.

I appreciate the graduate fellowship support from the Vodafone-US foundation through the Wireless Engineering Research & Education Center. Thanks to the Department of Electrical and Computer Engineering and the graduate school for their financial support.

I need to thank Dr. Tezaswi Raja from Transmeta Corp. for his help on starting this topic of research and providing me his logic simulator. Thanks to Anand, Alok, Raja, Lu for all the helpful discussions related to the research. Thanks to Kejun, Santosh, Manu, and Jin to provide me a refreshing working environment in that small office.

Finally and importantly, I would like to thank my parents for their support when I faced choices. A very special thank to my dear wife Yan, who has always been with me throughout all the challenges and struggles in my life, and without whose love, support, and inspiration, I would never be what I am today.

Style manual or journal used Bibliography follows those of the transactions of the Institute of Electrical and Electronics Engineers and is sorted in the alphabetic order.

Computer software used The document preparation package TEX (specifically LATEX) together with the departmental style-file `auphd.sty`.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Low power dissipation has become a crucial factor in the modern VLSI circuit design. With a rapidly evolving silicon technology, the transistor size keeps decreasing dramatically. While transistors are getting smaller and faster, power and *power density* (power per unit area) of VLSI circuits built with these transistors continues to become more serious. As the wireless communication, PDAs, mobile computing, sensor networks, etc., become popular, more mobile devices supplied by battery are widely employed. Larger power consumption impairs the device lifetime and some of its applications. For other devices not using battery as power supply, the increase of power density also brings challenges in power supply distribution and in removing the massive amount of heat generated by a chip.

The average power of CMOS device can be divided into *dynamic power* and *static power*. The major component of dynamic power is the *switching power* caused by signal switchings in a circuit; and the major component of static power is the *leakage power* caused by the leakage current of transistors. Dynamic and leakage power reduction are normally regarded as two separate problems, which require different analysis methods and approaches towards the solution. In this dissertation, we focus on the problem of dynamic (switching) power reduction in the context of increasing variability of process and circuit parameters as technology scales into the nanometer regime. We propose a linear programming based approach towards the solution.

1.1 Previous work

Power optimization techniques that concentrate on the reduction of switching power dissipation of a given circuit are called glitch reduction techniques. In conventional CMOS logic gates, the spurious transitions at the output due to the differential path delay are called *glitches* or *hazards*. The elimination of hazards has been widely discussed in previous books [33, 168, 174]. The principal idea is to find delay assignment for all gates in the circuit to reduce the differential path delays at gate inputs with respect to the inertial delays. The optimization techniques involved in hazard elimination are, *balanced delay* [14, 33, 104], *hazard filtering* [3], *transistor sizing* [25, 67, 179, 190, 219, 220], *gate sizing* [22, 23, 24, 60], and *linear programming methods* [4, 170, 171].

Balanced delay methods [14, 33, 104] equalize the delays of all paths incident on a gate and therefore need to insert *delay buffers* on selected fan out branches. Hazard filtering methods [3], utilizing the *glitch filtering effect* of gates, does not require the insertion of delay buffers. Transistor sizing methods [25, 67, 179, 190, 219, 220] have been proposed to optimize the power dissipation of a circuit by finding all transistor sizes. However, these techniques are limited by two major problems. First, transistor delay is not a linear function of transistor sizes. Second, transistor sizing techniques try to solve the problem in a large dimension of space by treating all transistors as parameters. Therefore, the global optimization of solution can not be guaranteed [184]. Gate sizing techniques [22, 23, 24, 60] reduce the complexity by modeling logic gates in a circuit as equivalent inverters, which are then scaled under sizing optimization. However, it still suffers from the nonlinearity problem of the delay model. Recently, Agrawal et al. proposed linear programming techniques [4, 170, 171] to derive the delay assignment in a circuit. In these approaches, circuit topology is used

to formulate a linear program and the delays of gates are treated as variables. The program returns the delay assignments under certain constraints and optimization objectives.

The linear programming based technique has the advantage that it only requires the modeling of the problem and leaves the optimization of the solution to the linear program solver. The best of all, it derives a globally optimal solution in a very short amount of time. Thus, it provides a convenient way of solving a large-scale optimization problem. However, previous works by Agrawal et al. [4, 170, 171] have some limitations due to the assumptions they make. In this dissertation, we propose to extend previous linear programming based optimization methods [4, 170, 171] considering the existence of process variations and a new aspect of input-specific optimization.

1.2 Motivations

1.2.1 Process variations

In previous LP models [4, 170, 171], each gate/buffer is assigned a fixed delay value and gate delays are adjusted precisely to satisfy the glitch filtering condition. However, in real integrated circuits, the delay of a gate can vary significantly from its expected value due to environmental factors (supply voltage V_{dd}, temperature T, etc.) and physical factors (process variations). The varying of the gate delays can easily alter the signal arrival times and corrupt the glitch filtering condition at a gate. The final switching power dissipation will deviate from its optimal value estimated under no process-variation. For large delay variations, glitch-filtering condition at every gate could be corrupted and the power saving by previous glitch reduction is severely degraded (as shown in Table 7.8 and Figure 7.7).

Note that process variation can lead to the variation of leakage power too. As technology scales, the leakage power component is getting significant, even becoming comparable to the dynamic power dissipation in some cases. Variation of leakage power could also affect the total power variation. In [200], the effect of process variation to major leakage reduction techniques (during functional mode) is discussed and compared. Results show that the worst-case leakage power (under variations) can be reduced to 2% of original value by those leakage reduction techniques when certain large delay penalty can be tolerated. Under a more stringent delay requirement, the worst-case leakage power and its variation can be reduced dramatically, i.e. 14% and 45% of the original values. Therefore, the impact of process-variations on leakage power can be small if certain leakage reduction technique is adopted. The reduction of leakage power variation during functional mode requires a separate investigation and we do not address it in our study.

1.2.2 Input-specific optimization

Previous LP modeling [4, 170, 171] considers the optimization of the circuit in the worst case. The LP solution ensures the absence of glitches for any input vector sequence. However, this constraint also imposes a burden on the design. For a circuit where the total propagation delay is restricted, the LP solution requires the insertion of lots of buffers. For conventional buffers, the total power dissipation of the circuit may not be minimal due to the increased power dissipation by those buffers. Even for the transmission gate type of buffer [169, 172, 173], which does not consume switching power, the total circuit area increases unnecessarily. The solution is thus not optimal. In reality, we may only optimize the circuit for certain input sequences that will be applied to the circuit, e.g., functional vectors a circuit receives while it is working. Optimization of the circuit for these vector

sequences ensures the low power dissipation when the circuit is in use and it can lead to a better solution because the optimization is more customized.

1.3 Problem statement

In this dissertation, we propose to consider delay variations during the optimization step and derive a robust solution. The problem to be solved in this work is: *finding a new linear programming method that removes most of the glitches in the circuit under the presence of process variation. That is, the optimized circuit maintains low power dissipation, which is not sensitive to the existence of gate delay variations. In addition, we find a method to reduce the trade-offs of the optimized circuit under a given input sequence.* Only the physical process variation is considered because it is the dominating factor that affects the delays [145].

1.4 Original contributions

In this dissertation, we propose a random delay model for gate delays in a circuit. Delays are modeled as random variables instead of deterministic values. We consider two basic types of process variations: inter-die variations and intra-die variations. We prove that the effect of the inter-die variation to the switching power dissipation is negligible. Based on the random delay model, we construct two new LP models for process-variation-resistant dynamic power optimization. Either the worst-case timing analysis or the statistical timing analysis is adopted in these two models. Our process-variation-resistant LP models can lead to a circuit that is more robust under process-variations. Both power dissipation and critical path delay maintain smaller deviations from their nominal values.

For input-specific optimization, we relax the constraints for certain gates where glitches are unlikely to occur. For a gate, certain input combinations are necessary for the production of a glitch at the output, which are noted as *glitch-generation patterns*. By observing the probability of glitch-generation patterns for each gate, we adaptively relax the glitch-filtering constraints. We are able to obtain a better solution with fewer buffer insertions and an almost identical power reduction as before.

1.5 Organization of the dissertation

The dissertation is organized as follows:

- Chapter 2 contains a survey of circuit level power optimization techniques, which are on the same level of design hierarchy.
- Chapter 3 has a survey of power estimation techniques.
- Chapter 4 illustrates the random delay model adopted in our new LP methods and our first process-variation-resistant LP model.
- Chapter 5 presents an improved process-variation-resistant LP model.
- Chapter 6 describes our input-specific optimization method.
- Chapter 7 gives experimental results for our process-variation-resistant LP models; the resulting designs are evaluated using Monte-Carlo simulation.
- Chapter 8 gives experimental results for our input-specific optimizations.
- Chapter 9 presents our conclusion and suggestions for future work.

Chapter 2

Circuit Level Low Power Techniques

2.1 Introduction

In this chapter, we survey low power techniques. We concentrate on device/circuit level techniques. For each technique, we introduce the idea and discuss its effectiveness on power reduction with examples. First, we address the low power problem and challenges. Then, we give an overview of low power techniques at architecture/logic level and device/circuit level. We then survey specific circuit level low power techniques and CMOS optimization techniques in detail.

2.1.1 Problem and challenges

Power dissipation problem

Along with the rapidly evolving silicon technology, the transistor size keeps decreasing dramatically. While transistors are getting smaller and faster, low power issue of VLSI circuits built with these transistors is getting more serious.

The average power of a digital CMOS device can be conceptually modeled as $P_{avg} = P_{static} + P_{dynamic}$ (see Section 2.2.1 for more details). The dominant component of P_{avg} so far has been the dynamic power which is composed of the switching power $P_{switching}$ (charging and discharging of the load capacitance) and the short circuit power P_{short} (when both P and N transistors are turned on during the switching). In the normal operation, switching power dominates the dynamic power and $P_{switching} = kC_L V_{dd}^2 f_{clk}$, where k is the

switching activity factor, C_L is the load capacitance, V_{dd} is the supply voltage and f_{clk} is the clock frequency.

It might appear that the switching power will decrease when transistor size decreases since the load capacitance is proportional to transistor size. However, the die area does not shrink with the technology. It actually means a 2X increase in number of transistors packed in a chip per generation [52]. At the same time, power dissipation of a single CMOS gate does not decrease fast enough to compensate for such an increase. The increasing power density (power dissipation per unit area) and related heat removal have been getting increasingly problematic.

Total capacitance. The minimum feature size (MFS) scales down by 30% every generation. If we consider a constant die area (with 2X more transistors), then the total transistor capacitance actually increases by 40% (for each transistor, 0.5X gate area, 0.7X gate oxide thickness). On the other hand, silicon technology is getting increasingly interconnect dominant. While gate capacitance for individual transistor decreases with the minimum feature size, interconnect capacitance per unit length decreases slower due to sidewall contribution. Then, there are more global/local interconnect layers due to the higher integration and complexity of a chip for every generation. Therefore, total capacitance of a chip keeps increasing.

Clock speed. The second fact contributing to the increasing of the power (density) is that clock frequency is scaled faster than the technology. The clock frequency has doubled with every technology generation in the past. This is a much faster scaling than the technology,

which should have been just 43% for constant power density [52]. The need for performance drives such scaling and makes the power density increase for each generation.

Leakage power. Third, as technology enters deep-submicron era, leakage current becomes more serious. Since the supply voltage decreases every generation, threshold voltage needs to decrease accordingly to avoid increased delay. However, the leakage current increases exponentially when threshold voltage decreases (see Section 2.2.1). The dramatically increased leakage current in the off state leads to the increase of static power P_{static}. The leakage current consists of both source-to-drain leakage due to sub-threshold conduction and drain-to-gate leakage due to electron tunneling across the ultra thin gate oxide. The latter is not yet a significant portion of leakage, but it is getting more noticeable and could dominate as technology scales further. The increase of leakage current has a direct impact on the standby power consumption of a chip.

Architecture trend. Another important factor that contributes to the increasing power density is that the hardware architecture trend is toward more flexible (programmable) and reusable cores. Comparing to application specific architecture, it is much less energy efficient. Programmability is a common requirement for designing a large scale SoC (system-on-chip). It ensures function flexibility, and helps in post-fabrication bug fixing and tuning. However, flexibility and programmability impose an energy burden for the chip. The power-performance ratio required by a processor to carry out a given task can be several order of magnitude higher than that achieved by an application specific architecture [164].

Challenges and solution

It is obvious that the power issue has become an increasingly important factor in VLSI design. As the wireless communication, PDAs, mobile computing, sensor networks, etc. become popular; mobile devices supplied by battery will be widely employed. Increases of switching power and leakage power consumption have direct impact on the battery life. For some of those devices (e.g., sensors in a sensor network), it is even difficult to frequently replace the battery. Higher power consumption impairs their lifetime and application. For other devices not using battery as power supply, the increase of power density adds difficulties to the power supply distribution and thermal management to remove the massive amount of heat generated by a chip.

Numerous low power techniques have been developed since last decade. They can be classified into device, circuit, logical and architecture levels. For example, transistor sizing is a device level technique that optimizes the size of transistors in a circuit. Different circuit design styles like dynamic logic, pass transistor logic, etc., are circuit level techniques. Power optimized synthesis of logic structure is a logic level technique. Instruction set optimization to reduce the switching ratio is a good example of architecture level technique. Logic/architecture level low power techniques have a significant influence on the total power consumption of a system. However, device/circuit level techniques are more fundamental and can always be applied with any logic/architecture level technique. In this chapter, we concentrate on device/circuit level techniques and give a survey of those techniques. For each technique, we introduce the idea and discuss its effectiveness in power reduction.

2.1.2 Low power techniques

Architecture/logic level

One approach to power reduction at the architecture level is to build a power manageable architecture so that one can eliminate idle power consumption (power consumed when the hardware is not in use), and run time slack by controlling the clock activity, voltage, frequency, and even the device threshold voltage. Reader can refer to recent papers [15, 17] for surveys of power manageable architecture techniques, which consist of power manageable hardware, power management software and system level Dynamic Power Management (DPM) techniques. Here we will only give few examples and illustrate the basic ideas.

One way to manage power at the architecture level is to use multiple voltages and clock frequencies. In multiple-voltage circuits, two or more supply voltages are distributed on chip according to the criticality of the path. Time-critical paths are supplied by a higher voltage and a lower supply voltage is used to reduce the power for non-critical paths. In variable-voltage circuits, supply voltages are modulated during the system operation. It is a very powerful technique because it can trade off power for speed at run time to fine tune performance and power according to the workload. In practice, however, it requires smart design techniques because voltage change requires non-negligible time and clock speed must be varied accordingly when supply voltage changes.

Clock gating is another common power management technique that allows turning off clock for idle modules in a circuit. Power savings are achieved in the registers (whose clock is halted) and in the combinational logic gates where signals do not propagate due to the freezing of data in registers. Clock gating is widely used because it is conceptually simple, has a small overhead in terms of additional circuitry and often has a zero performance

overhead because the component can transit from an idle to an active state in one (or few) cycles. The main design challenges in the implementation of clock gating are: 1) to construct an idleness-detecting circuit which is small and accurate and 2) to design gated-clock distribution circuitry that introduces minimum routing overhead and keeps clock skew under a tight control [151].

Leakage is a major concern in idle-power consumption. Most leakage-reduction techniques, e.g., dual-Vt, Variable Threshold CMOS (VTCMOS) and power supply gating, etc., can be considered as architectural level power management techniques. For the dual-Vt technique, the basic idea is to use low threshold transistor (fast and leaky) on time-critical paths and high threshold transistor (slow and less leaky) on non-critical paths. The dual-Vt technique tends to lose its effect when more paths become critical. VTCMOS allows dynamic control of threshold voltage via substrate biasing. It has a better leakage reduction effect than the dual-Vt but requires standby control circuit to detect the idleness of a module and then apply the biasing. Finally, the ultimate solution to avoid leakage is to shutdown the power supply during the standby time. An advantage of this approach is the wide applicability to all kind of electronic components, i.e., digital and analog units, sensors, and transducers. A major disadvantage is the wake-up recovery time, which is typically higher than in the case of clock gating because of the re-initialization of components.

Yet another approach for architecture level power reduction is the application dependent specialization [15], which is an ad-hoc way to specialize hardware platform for an application without compromising the reuse and design flow streamlining. Readers can refer to recent books and survey papers [118, 157, 164] for more details.

Device/circuit Level

We listed the specific low power techniques in Table 2.1 and power optimization techniques for CMOS in Table 2.2. Most of these technologies/techniques are at device/circuit level. However, we included leakage reduction techniques because they are getting increasingly important now. In this section, we will give a brief summary for each of them. Detailed discussion is provided in later sections.

Low Power Techniques		**Switching Power**	**Short-circuit Power**	**Leakage Power**	**Static Power**	**Power Reduction**
CMOS		Yes	Yes	Yes	No	Nanowatt standby power [214]
Domino CMOS		Yes (no glitches, has wasted discharges)	Maybe (caused by the contention)	Yes	No	128% speed up with 41% more power consumption [48].
PTL	**Pure PTL**	Yes (reduced by a smaller capacitance)	Almost no (only exist at output inverter)	Yes	Maybe (due to V_{th} drop)	30-50% reduction by [1, 116, 156, 158, 227, 228], 44% of the power-delay product in [176]
	Mixed PTL/ CMOS	Yes (reduced)	Yes	Yes	Maybe (due to V_{th} drop)	more than 20% reduction in [41], about 50% of power delay product in [42]
Asyn. Design	**Self-timed Logic**	Yes (glitches eliminated)	Maybe (caused by contention)	Yes	No	25% reduction in [108]
	Others	Yes (no clock signals and clock networks)	–	Yes (reduced by a shorter signal quiescent time)	–	5 times saving with 20% area overhead in [207], up to 5 times less power in [149, 150]
Adiab. Switch. & Engergy Recov.	**Fully Adiab.**	No (asymptotically zero)	No	No (negligible)	No	up to 6 times less energy per addition CLA in [122]
	Partial Adiab.	Yes (asymptotically nonzero)	No	No (negligible)	No	10-20 times power gain inverter chain in [57, 127, 138]

Table 2.1: Low power technologies: specific technologies and theirs power components. Except the CMOS technology, power reduction by each technology is compared to the static CMOS (clocked) counterpart.

Low Power Technology			Basic Idea	Power Reduction Effects
Dynamic Power Reduction	**Transistor/Gate Sizing**		Size the transistor/gate to optimize switching power and short-circuit power under the delay constraint	40% to 50% by TILOS [66], 50% to 65% reduction in [79, 223]
	Glitch-reduction Techniques	**Balanced Delay**	balance differential path delay	30% power reduction in [104]
		Hazard Filtering	Increase gate inertial delay to filter out glitches	27% power reduction in [4]
		Transistor Sizing	size the transitor/gate to balance the path	32% to 46% reduction in [220]
		Linear Programming	Use linear programming to derive gate delay assignments	up to 62% reduction in [170], 25% to 54% reduction in [171]
Leakage Reduction	**Input-vector Control**		switch input vector to low leakage pattern during standby	up to 2x leakage reduction [229]
	Body-bias Control	**VTCMOS**	dynamically change threshold voltage to high Vt at standby	100μA I_{leak} at active mode; 0.1μA I_{leak} at standby [115]
		DTCMOS	the floating body and the gate of a high Vt transistor are tied together	up to 5.5 time higher current drive when device is on (low Vt) [7]
		Dual-Vt	Low Vt for critical path; High Vt for non-critical paths	up to 80% leakage reduction [215]
	Power-supply Gating		Insert "sleep" transistor at pull down path or use power supply regulator to turn off the power supply during standby	virtually 100% Leakage reduction [59]

Table 2.2: Low power technologies: optimization techniques for CMOS circuits.

Complimentary MOS (CMOS) was first proposed by Wanlass and Sah in 1963 [214]. The CMOS process is more complex than the NMOS process because it provides both n-channel and p-channel transistors on the same chip. However, CMOS circuits can achieve low power consumption by eliminating (most if not all) static power.

Domino CMOS is a dynamic logic family originally suggested in [111], which combined the speed and power advantage of the dynamic logic circuit and the stability and ease of use of static logic (full Complementary MOS) circuit. Compared to static CMOS, domino CMOS reduces the dynamic power because it has a smaller switching capacitance (having fewer transistors), no spurious transitions (glitches) and no short circuit current as in CMOS. However, domino CMOS does have some serious drawbacks that lead to additional power

consumption. One problem is the so called "contention" (see details in Section 2.2.2), which can consume additional power. In addition, the operation of domino CMOS requires pre-charge and evaluation phases, which means some nodes are charged and discharged unnecessarily. Overall, domino CMOS still appears to have a better time power trade-off than the static CMOS.

The difference between Pass Transistor Logic (PTL) and CMOS logic is that the source of the pass transistor network is connected to some input signal instead of the power lines and ground. Pass transistor logic is attractive because it can reduce the number of transistors in implementing XOR gate, multiplexers, registers, and other key building blocks. However, the threshold voltage drop at the output requires level restoration, which means extra circuitry must be added. There are still debates about the power efficiency of PTL and CMOS. In practice, application of PTL/CMOS mixed logic has achieved considerable power reduction.

Self-timed Logic is an asynchronous design that utilizes handshake signals to synchronize the data exchange between asynchronous elements. One major advantage of self-timed logic is the elimination of the clock generator and distribution network, which could otherwise consume a significant portion of power. In addition, self-timed logic inherently powers down the unused modules and saves power consumption by them. One disadvantage of self-timed logic is that for certain logic families it may suffer from the "contention" problem as in domino CMOS. With the requirement of dual-rail encoding (for completion signal), the energy consumption per transition could be high, which limits its application in a continuously active data path.

Except for self-timed logic, asynchronous designs have recently drawn resurgent attention because of their low powers feature. A major advantage of an asynchronous design

is that it does not require a power consuming clock network. New design technique (e.g., Tangram framework [100]) supports the plug and play composition of asynchronous components into systems, which significantly simplifies the design task for a large system. Many asynchronous designs exhibit dramatic power reduction especially for applications that have significant computation load fluctuation and large disparity between average performance and peak performance, e.g., general purpose microprocessors, error correctors, etc.

The fundamental cause of energy dissipation in a CMOS circuit is the charge transportation from V_{dd} to load capacitance and to GND. The principal idea of adiabatic switching is to minimize the energy dissipation during this process by slowing down the charging/discharging operation. Combining with reversible computation (no information loss during computation), one can build a "fully adiabatic" circuit, which has asymptotically zero power consumption. The limitation of fully adiabatic circuit is that the function of the circuit has to be reversible, which limits its application. "Charge recycling" or "energy recovery" are terms used more recently for describing circuit techniques that do not require reversible logic but "recycle" the information representing charges and use adiabatic switching to reduce the energy dissipation. In practice, the power saving is dramatic, sometimes as high as one order of magnitude. However, the major drawback of these techniques is that the operating frequency cannot be very high (due to the adiabatic switching principle).

As CMOS was prevailing in the last few decades, numerous low power techniques have been proposed to enhance the power performance of CMOS based circuit. Transistor/gate sizing is a technique determining the sizes of transistor/gate in a circuit. In the past, optimizations were primarily for circuit delay and area. With the growing concern for low power dissipation, new transistor/gate sizing techniques for power optimization have been proposed. Glitch (spurious transitions before a signal reaches the steady state value)

reduction is another important topic in CMOS low power design. To eliminate glitches, the basic idea used is to balance the paths (path balancing) and/or filter the glitch by gate inertial delay (hazard filtering). Transistor/gate sizing can be used for the optimization. One can also use linear programming algorithms to derive the delay assignments in the circuit and then realize these delay assignments by buffer insertion or gate level design.

At last, leakage reduction techniques have been proposed to reduce the static power in CMOS circuit during the idle time. Except for the techniques mentioned in the previous section, there are additional techniques like input vector control, DTCMOS (Dynamic Threshold CMOS), etc. We will give a detailed discussion in Section 2.3.2.

2.2 Specific technologies

2.2.1 CMOS technology and power components

Complimentary MOS (CMOS) was first proposed by Wanlass and Sah in 1963 [214]. It was then developed for commercial use in the early 1970s. CMOS logic was originally thought to be too complicated, expensive, and slow compared to the NMOS technology. It was also prone to a failure mechanism called latch-up [134], which effectively shorts across power supply on the IC and is highly likely to cause irreparable damage. However, with the improvement of technology and the increasing importance of power dissipation as ICs were getting larger, CMOS almost completely replaced the NMOS technology. The major power reduction by CMOS is due to the elimination of the static power. Since most low power techniques discussed in this chapter are compared to or related to CMOS technique, we need to understand the power components in CMOS circuits first.

There are four sources of power dissipation in a CMOS circuit, which can be summarized in the following equation:

$$P_{avg} = P_{switching} + P_{short-circuit} + P_{leakage} + P_{static}$$

Switching power

$P_{switching}$ represents the switching component of power and $P_{switching} = k \cdot \frac{1}{2} C V^2 f_{clk}$, where k is switching activity factor of the node (average number of transitions in one clock period). Figure 2.1 shows the charge flow at the output of a simple inverter.

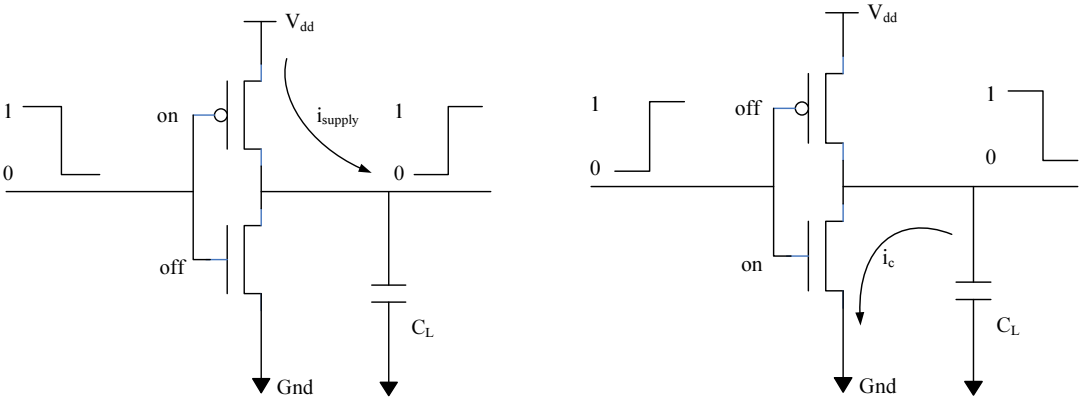


Figure 2.1: The charge flow for a simple inverter: (a) charging of the load capacitance, (b) discharging of the load capacitance.

When output of a CMOS gate makes a 0 to 1 transition, the energy drawn from power supply is

$$E_{supply} = \int_0^T P(t)dt = V_{dd} \int_0^T i_{supply}(t)dt = V_{dd} \int_0^{V_{dd}} C_L dV_C = C_L V_{dd}^2$$

where T is the charging time, C_L is the load capacitance (an abstract node capacitance that consist of gate capacitances, interconnect capacitances and the diffusion capacitances [33]). The energy stored in the load capacitance is

$$E_C = \int_0^T P_C(t)dt = \int_0^T V_C i_C(t)dt = \int_0^{V_{dd}} C_L V_C dV_C = \frac{1}{2} C_L V_{dd}^2$$

Therefore, half of the energy drawn from the power supply is dissipated during the charging through the PMOS network. Similarly, the remaining half of the energy stored in the capacitance will be dissipated through the NMOS network during 1 to 0 transition at the output.

Short-circuit power

$P_{short-circuit}$ represents short-circuit component of power, which occurs during a short period of time when the input switches and both PMOS network and NMOS network are ON. A direct current path between V_{dd} and GND exists during that period. Unlike the switching component that is independent of the rise and fall time at the input of a logic gate, short-circuit component is very much affected by the rising and falling time of input signals. The short-circuit current is significant when the rise/fall time at the input of a gate is much longer than the output rise/fall time [33]. Therefore one approach to minimize short-circuit power is to make the output rise/fall time larger than the input rise/fall time. However, this will slow down the circuit and might cause short-circuit current in fan-out gates. Therefore, there is no simple answer to this problem and most chip designers eventually retreat to some design rules. It has been shown, by sizing transistors for equal rise and fall times, the

short-circuit component can be kept less than 20% of the dynamic power [211]. Figure 2.2 shows the short-circuit current during the input transition.

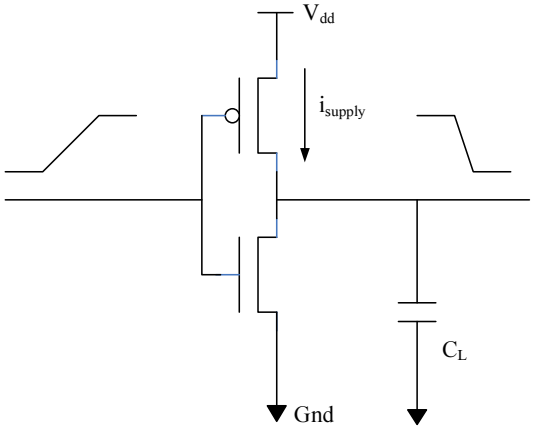


Figure 2.2: The effect of slow rise/fall time at the input of a gate on short-circuit power dissipation.

Leakage power

For leakage component $P_{leakage}$, there are three types of leakage currents, diode leakage, sub-threshold leakage and direct-tunneling current. Figure 2.3 shows these three types of leakages.

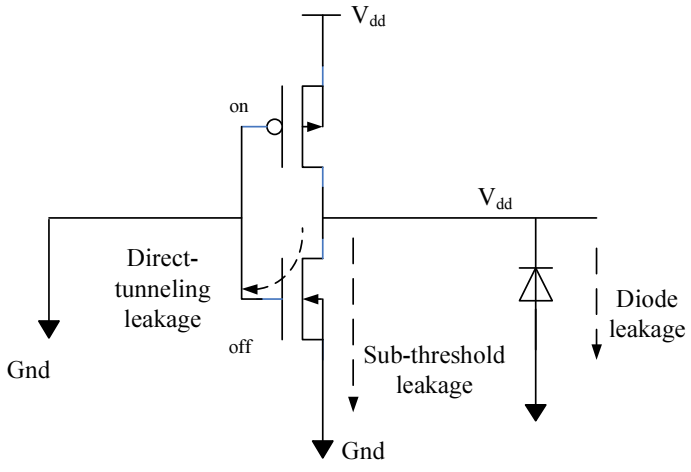


Figure 2.3: Three types of leakage currents in a CMOS inverter.

Diode leakage occurs when a transistor is turned off and another active transistor charges up/down the drain with respect to the former's bulk potential. Consider the inverter in Figure 2.3, the NMOS transistor is turned off while PMOS transistor is on. The drain-to-bulk voltage for the NMOS transistor is V_{dd}, which reverse biases the drain-to-bulk diode. The leakage current for the diode is given by [33]:

$$I_{diode} = I_s(e^{\frac{V_{db}}{V_T}} - 1)$$

where I_s is the reverse saturation current, V_{db} is the diode voltage, and $V_T = KT/q$ is the thermal voltage.

Another leakage component is the sub-threshold leakage, which occurs due to carrier diffusion between the source and the drain when the gate-source voltage is below the threshold voltage and carrier drift is dominant. In this regime, the MOSFET behaves similar to a bipolar transistor and the current in the sub-threshold region is given by [192],

$$I_{sub-Vt} = Ke^{\frac{V_{gs}-V_t}{nV_T}}(1 - e^{-\frac{V_{ds}}{V_T}})$$

where $K = I_{DS0}W/L$ is a function of technology (I_{DS0} is a measured constant), V_T is the thermal voltage, V_t is the threshold voltage, and n is the slope parameter. For $V_{ds} >> V_T$, $(1 - e^{-\frac{V_{ds}}{V_T}}) \approx 1$.

As technology scales down, gate oxide thickness has to be scaled down accordingly to minimize the degradation of device behavior. The International Technology Roadmap for Semiconductors (IRTS) predicts that the gate oxide thickness will reach $1nm$ as early

as 2006 [88]. This low oxide thickness gives rise to high electric field, resulting in considerable direct-tunneling current. Gate direct-tunneling current is due to the tunneling of electrons (or holes) from the bulk silicon through the gate oxide potential barrier into the gate [194]. For CMOS devices with larger oxide thickness, major leakage mechanism is the sub-threshold current. However, in the ultra-thin gate oxide regime, gate tunneling current becomes appreciable and dominates the total "off" state leakage current of the transistor [226]. The direct-tunneling is modeled as [181]:

$$J_{DT} = A(V_{ox}/T_{ox})^2 e^{\frac{-B(1-(1-V_{ox}/\phi_{ox})^{3/2})}{V_{ox}/T_{ox}}}$$

where J_{DT} is the direct-tunneling current density, V_{ox} is potential drop across the oxide, ϕ_{ox} is the barrier height of tunneling electron and T_{ox} is the oxide thickness. A and B are physical parameters [181]. The tunneling current increases exponentially when oxide thickness decreases.

Static power

Static power (P_{static}) refers to the power consumed when signal is at the steady state other than the leakage power. Compared with NMOS logic, CMOS logic only consumes power at switching. There is no steady current drawn from the V_{dd} to GND. Therefore, CMOS circuit normally does not have this power component. However, in some cases, CMOS circuit might still consume static power. Considering a PTL/CMOS mixed circuit as shown in Figure 2.4, when input of a CMOS inverter is connected to the low output of an NMOS pass transistor. The threshold voltage drop at the NMOS pass transistor output makes the input of the inverter to equal the threshold voltage V_{th}. The resulting inverter

has a partially "on" NMOS pull down transistor and forms a static current path from V_{dd} to GND. This static power might be significant if the circuit is idle most of the time.

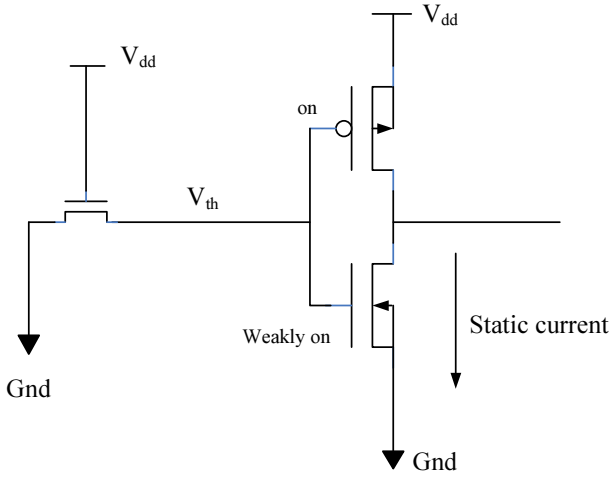


Figure 2.4: Possible static power dissipation of a CMOS inverter.

2.2.2 Domino CMOS

Basic idea

Domino logic was originally suggested in [111], combining the speed and power advantage of dynamic logic circuits and the stability and ease of use of static logic (full complementary MOS) circuits. Figure 2.5 shows a typical n-type domino circuit. A static invert is added after the dynamic stage to solve the cascading problem in dynamic logic circuits [206]. Because the evaluation of a stage begins after the evaluation of the previous stage is finished and the voltage dropping at the dynamic points Y resembles domino style falling, it is given the name "domino" logic.

The operation of a domino logic circuit has 2 phases. For an n-type domino, the precharge phase begins when CLK is low. Q_p is turned on and the dynamic point Y is charged

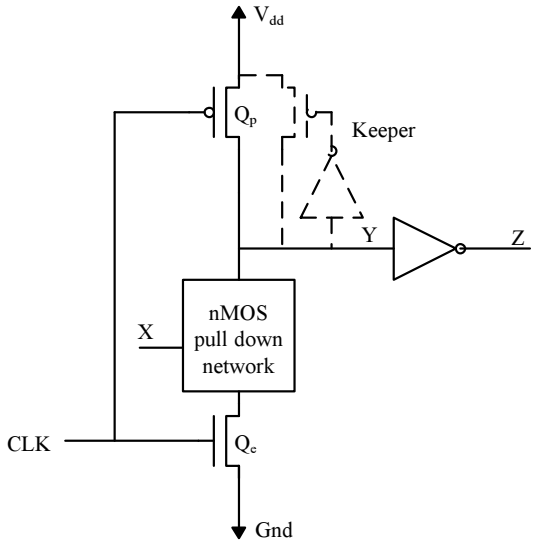


Figure 2.5: A typical n-type domino logic circuit. A keeper is drawn in dashed line.

to V_{dd}. When CLK is high, Q_e is turned on and it enters the evaluation phase. The NMOS pull down network is turned on/off depending on the input X. Y is discharged if the pull down network has a discharge path. The output Z is then evaluated consequentially.

Advantages

Comparing with the static CMOS logic, domino logic reduces the dynamic power consumption in several ways. First, it eliminates the spurious transitions in static CMOS logic circuits. Spurious transitions are multiple transitions in the output signal before it settles to the correct logic value. To be more specific about the magnitude of this problem, an 8-bit ripple-carry adder with a uniformly distributed set of random input patterns will typically consume an extra 30% in energy [34]. Domino logic circuits have at most one power-consuming transition per clock cycle and thus inherently do not have such a problem.

Domino logic has a much smaller parasitic capacitance than static CMOS circuits because of the elimination of one PMOS (or NMOS) network. It typically uses fewer

transistors to implement a given logic function, thus has a smaller switched capacitance (faster operation speed) and smaller dynamic power consumption per transition.

Domino logic does not have the short-circuit current as in static CMOS circuits. In CMOS, short-circuit current exists when both PMOS and NMOS networks are conducting. Because domino logic only has one pull down (or pull up) network, it is normally not subject to such a problem.

Disadvantages

While domino logic has many advantages over the static logic circuit in terms of dynamic power consumption, it has severe drawbacks that could lead to additional power consumption. For example, some additional transistors are needed to construct a "keeper" to insure the charge sharing on the pull down NMOS network does not lead to significant voltage drop at the dynamic point Y, which can result in a wrong logic evaluation. Simple design of "keeper" as shown in Figure 2.5 will cause "contention", which means both the pull up transistor for dynamic point Y and the pull down network are turned on at the same time (at the beginning of evaluation phase). Contention leads to additional power consumption as the short-circuit power consumption in static CMOS. More advanced keepers have been proposed [5, 62] to eliminate the contention and achieved as much as 10-20% of the dynamic power reduction.

One serious drawback of domino logic is the existence of the pre-charge phase. Even if inputs of the circuit have no change for a long time, the circuit still consumes dynamic power for every clock cycle. That means some nodes are charged in pre-charge phase and then discharged immediately in the following evaluation phase. Therefore, domino logic exhibits a much higher activity rate (the probability of transitions) than static logic circuits. As

described in [34], assuming random input, a dynamic NOR gate has an activity rate of 3/4, while a static NOR gate only has an activity rate of 3/16. In addition, the clock buffer that drives the charging of pre-charge transistors consumes additional power.

Lastly, the power-down technique widely used in static logic circuits that disables the clock for those idle modules is not very effective for domino logic. To maintain the state of the domino logic during the sleep mode, some additional circuitry must be added, which results in a slightly higher parasitic capacitance and slower speed [34].

Discussions

Because of the two major drawbacks of domino logic, the high activity factor and the power consumption by the clock network, domino logic is less power efficient than static logic in some cases. In [109], a comparison is made for a CLA (carry look ahead) adder circuit implemented with static CMOS and domino logic. A 32-bit CLA implementation in domino logic has just a 24% improvement of delay than the static CMOS implementation but requires more area (19%) and significantly larger amount of power (140%).

However, in some other case [48], domino logic still exhibits the best trade-off between area, time, and power. In [48], the 64-bit RCA (ripple carry adder) based domino logic square-rooting array speeds up the corresponding RCA-based CMOS implementation by about 128% and only consumes 41% more power.

2.2.3 Pass transistor logic

The type of logic style used in logic gates influences the power dissipation of the circuit. Total load capacitance is a function of the number of transistors in a circuit. One approach

to reduce power is to use pass transistor logic (PTL) over the conventional CMOS logic. Figure 2.6 shows an example of AND/NAND gate in PTL.

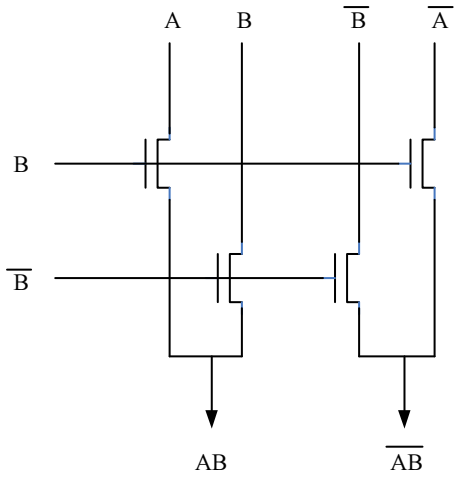


Figure 2.6: Example of PTL: an AND/NAND gate in the CPL (Complementary Pass-transistor Logic) family.

Advantages

The difference between PTL and the conventional CMOS logic is that the source of the pass transistor network is connected to some input signal instead of the power lines and ground. The advantage of PTL is that one pass transistor network is sufficient to perform the logic operation, while conventional CMOS logic always requires both NMOS and PMOS networks. Pass transistor logic is attractive because it can reduce the number of gates needed for implementing XOR gate, multiplexers, registers, and other key building blocks [34]. The efficient implementation of an XOR gate is especially important because it is the essential element in most arithmetic functions, such as adders and multipliers. Various investigations of pass transistor logic with respect to low power dissipation have been carried out [1, 116, 156, 158, 227, 228]. According to these investigations, CPL and

other pass transistor logic style reduce the power by 30-50% compared to their conventional CMOS counterparts.

Numerous pass transistor logic styles have been proposed since 1990s, including CPL [228] (Complementary Pass-transistor Logic), LEAP [227] (LEAn integration and Pass transistors), SPL [193] (Single-rail Pass-transistor Logic), SRPL [158] (Swing Restored Pass-transistor Logic), DPL [152] (Double Pass-transistor Logic), DPTL [159] (Differential Pass-Transistor Logic), EEPL [187] (Energy Economized Pass-transistor Logic), PPL [156] (Push-pull Pass-transistor Logic), etc. Most of these pass-transistor logic families use a dual-rail structure, while LEAP and SPL use single-rail structure to reduce the number of transistors. In [176], the SPL implementation of a 16-bits multiplier achieves 56% reduction of power-delay product as compared to the CMOS implementation.

Disadvantages

PTL has its own disadvantages and is not always better. One disadvantage of PTL is the threshold voltage drop through the NMOS transistor while input is "1", which makes level restoration at the output necessary. The level restoration circuitry avoids the static current in the subsequent output inverter or logic gate but adds some additional overhead to the circuit. In order to decouple gate inputs and to provide acceptable driving capabilities, inverters are often attached to the gate output. Another problem is that logic function implemented using PTL must be in a multiplexer structure, which limits the number of logic functions that can be implemented efficiently. Some simple gates may not be implemented very efficiently using PTL.

Comparison with CMOS

There are evidences that the low power advantage of PTL over conventional CMOS is over-estimated in the literatures. In [234], different PTL families and the conventional CMOS logic are compared with three sets of logic functions, full adder, multiplexer and some other simple gates. Surprisingly, results showed that CMOS logic has a significant margin over PTL logic in most cases except for the full adder. The power reduction by CMOS logic over PTL can be as high as a factor of 3 (with only 20% speed degradation). More recently, another study on the behaviors of conventional CMOS and CPL full adder circuits [167] gives us more insight into the advantages and disadvantages of CPL and conventional CMOS. This study shows that a full adder with minimum power consumption can really be achieved by conventional CMOS design style. However, the minimum delay full adders are obtained with CPL. Therefore, the comparison between these two depends on the choice of the design point on the power-delay curve.

Mixed PTL/static logic

Although there is still controversy about the choice of PTL or CMOS, mixed PTL/static logic has been proposed [40, 91, 224, 225]. Unlike conventional PTL design where dedicated buffer are inserted in pass transistor tree to restore the drivability, mixed PTL allocates certain number of static gates at optimal locations within pass transistor tree to boost the drivability as well as perform the logic function. Therefore, the performance and power consumption of the mixed PTL circuit are further improved. Results in [41] show that a mixed PTL achieves at least 20% power reduction compared to its pure static CMOS counterpart. The impact of technology scaling on mixed PTL circuits is studied in [42]. Technologies of $0.18\mu m$, $0.13\mu m$ and $0.1\mu m$ were used in the study with V_{dd} scaled accordingly. Results

show that a mixed PTL circuit is robust against the technology scaling and maintains its advantage over the conventional static CMOS (by about 50%) and by big margin over the dynamic logic.

2.2.4 Self-timed logic

Basic idea

Self-timed logic is an asynchronous design method. It provides a way to design asynchronous logic circuits such that their correct behavior depends neither on the speed of their components nor on the delay along the communication wire. An extensive discussion of self-timed logic and its advantage over synchronous designs are provided in [182].

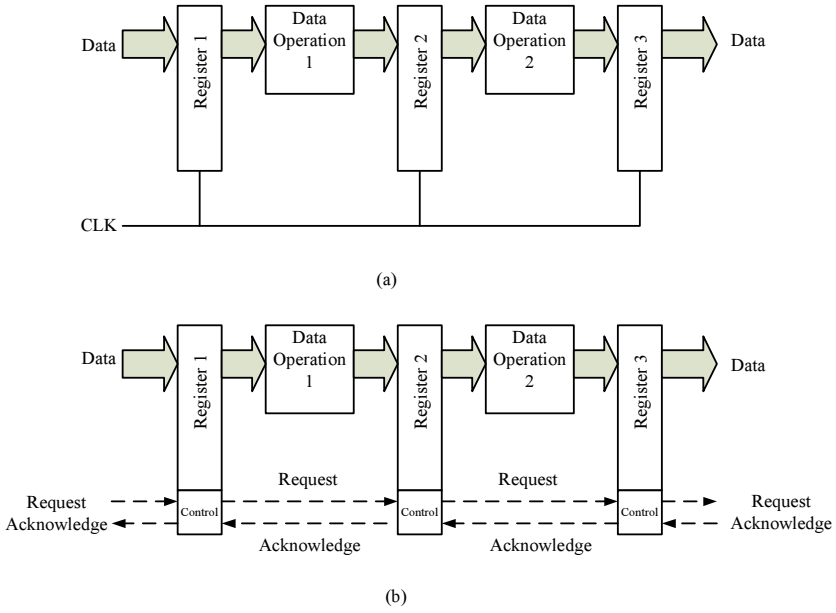


Figure 2.7: Illustration of simple pipelines of logic: (a) clocked pipeline, (b) self-timed pipeline.

One of the key concepts in self-timed logic is the use of a handshake signal [191] to synchronize the data exchange between asynchronous elements. Figure 2.7 shows a synchronous

pipeline and an asynchronous one using handshake signals [165]. For a synchronous pipeline as shown in Figure 2.7(a), the clock period has to be chosen longer than the worst-case delay of any computation element in the pipeline. However, in the case of the self-timed logic (Figure 2.7(b)), clock is removed and data transfer is controlled by the request/acknowledge signal passing between stages. Data will be accepted in a register as soon as the current data has left and new data from previous stage is available. Data will be transferred to the next register as soon as it is available and the next stage is ready to accept it.

Advantages

One advantage of self-timed logic is that it inherently powers down unused modules [34]. In synchronous designs, the logic between registers keeps performing computation as long as inputs change. However, some of such computation may not be "useful" and consumes unnecessary energy. To detect and power down those unused module, it requires special design effort and power-down circuitry in synchronous designs. However, such power-down of unused modules is inherent for self-timed logic, since transitions happen only when requested. Furthermore, the elimination of power-consuming clock drivers gives self-timed circuit an additional edge in power efficiency.

Disadvantages

Self-timed logic requires the generation of a completion signal to indicate its output is valid. This requirement adds some overheads in circuitry and power consumption. There are several circuit approaches to generate such completion signal. A common method is to use dual-rail encoding [51], which utilizes two signals (complement to each other) to represent one logic output. For certain logic families such as DCVSL (differential cascode

voltage switch logic) [43, 90], dual-rail encoding is implicit. The completion signal of a DCVSL gate is just a simple ORing of the outputs, which has only a small overhead. DCVSL families are similar to domino logic since they all have pre-charge and evaluation phases. However, DCVSL gates are not driven by a single clock signal but by the completion signals. For each computation of a DCVSL gate, dual-rail coding guarantees a switching event because the pre-charge step sets the completion signal to low. Like the domino logic, DCVSL can have the "contention" problem and may consume additional power. It was found that the dual rail DCVSL family consumes at least twice the energy per transition than a conventional static family [34]. Therefore, self-timed implementation may not be power efficient for data path that is continuously active.

Discussion

Self-timed logic can be used to eliminate the spurious transitions caused by dynamic hazards [76], which are inherent problems for static logic designs. The unnecessary switching can consume 30% [112] more energy than is required by the computation. Dynamic logic, such as domino logic, could be a solution because it only has at most one transition for each clock cycle. But domino logic has significant overhead [109] due to clock signal loading, clock driver, and high activity factor. As shown in [46], self-timed logic can be used to remove the spurious transitions from the functional level. Furthermore, an improved design in [108] achieves at most 25% power reduction when compared to a static logic design.

2.2.5 Asynchronous design

Advantages

In the past decade, there has been a resurgence of interest in asynchronous logic design, which has been largely neglected in previous decades. One reason is that synchronized circuits have begun to encounter some serious limits. As VLSI technology keeps adding more transistors on a single chip, the difficulty to maintain global synchronization, on which synchronized circuit depends, is increasing. Clock skew is already a problem at the board level and increasingly becoming a problem on a single chip [161]. Asynchronous logic is not affected by clock skew because it does not require a global synchronization.

Compared to asynchronous logic, clocked synchronous logic has disadvantages in low power application:

- Each register consumes power in every clock cycle, regardless of the change of the state. If dynamic logic is used, then each combinational logic module consumes power in every clock cycle.

- Clock distribution and generation network consumes significant amount of power. In a high performance processor, it can reach 40% of the overall power consumed [77].

- The clock period is chosen to satisfy the worst-case scenario, which means some modules become quiescent well before the end of a clock period. The leakage power consumed by modules in the quiescent state is an increasing problem in the deep sub-micron technologies.

Asynchronous logic was abandoned before because of its inherent difficulty with large-scale design. However recent advances in design methodology [100, 163] have solved many

early problems and demonstrated the efficiency of asynchronous logic in low power applications.

New techniques

The Tangram framework [100] by Philips Research Labs pioneered the concept of VLSI programming in which the behavioral description of a design is specified in a high-level design language called Tangram. A so-called silicon compiler has been implemented that translates Tangram programs into asynchronous circuits [209]. Tangram uses handshake signaling as the asynchronous timing discipline because it supports the plug and play composition of components into systems. The alternative to handshake signaling would be to compose asynchronous finite state machines that communicate using the fundamental mode or burst-mode assumptions. However, attempts to use this path to design industrial circuits have suffered from severe reliability and interface problems [45]. Tangram framework achieves as much as 5 times power saving when compared to a synchronous version employing clock gating technology and has only 20% area overhead [207]. Tangram system has been embedded into a commercial CAD environment (Cadence) by the OMI-EXACT project [64], allowing a user to optimize certain specialized circuit elements for higher performance in area, power and speed.

Applications

Asynchronous design is not for all applications. It is only suitable for applications that have significant computation load fluctuation and large disparity between average performance and peak performance, for example, general-purpose microprocessors, and circuits like error correctors, which are operational for small amount of time. One such example is

Reed-Solomon error correctors operating at audio rates [207], as demonstrated by Philips Research Labs. Later, in [208] two different asynchronous realization of this decoder are compared with a synchronous version. The single rail realization consumed five times less power than the other. The filter bank for a digital hearing aid [149, 150] was another successful demonstration. The asynchronous implementation results in a factor of five less power consumption. Other applications include an infrared communications receiver IC [133], pager subsystems [99], DSPs [87, 103], and cryptographic ASICs [121, 185].

Several groups have exploited the low power potential in using asynchronous logic to build programmable processors. In such a scenario, the computation load could vary significantly. AMULET2e of University of Manchester is an embedded system chip incorporating a 32-bit ARM compatible asynchronous core, a cache and several other system functions [69, 70, 73]. The synchronous versions of ARM are already well known for their low power consumption. Thus, the reduction in power per MIPS is modest. However, the absence of a high-frequency oscillator and Phase-Locked-Loop (PLL) offers a unique combination of features in the asynchronous idle mode. Since there is no need to deal with the slow restart and stabilization problem associated with the oscillator and PLL, it has a 3μW idle power consumption and an instant response to an external interrupt. A more complex AMULET3 [72] has also been developed. Although the initial implementation does not demonstrably beat ARM9 on performance and power due to limited development effort and experience, it shows that asynchronous implementation can have great potential in competing with its synchronous counterpart even for such a large and complex system. A self-timed data-driven multimedia processor [110, 195, 196] was designed by Sharp Corporation and the Universities of Osaka and Kochi. With eight programmable, data-driven processing elements, this processor targets applications including future digital television receivers. It

has an impressive peak performance of 8600 MOPS with power consumption below 1W (at $0.25\mu m$ CMOS, 2.5 V supply voltage). Another example is the 80C51 microcontroller redesigned by Philips Semiconductors together with Philips Research. This asynchronous version [210] consumes about four times less power than its synchronous counterpart. Cogency redesigned a programmable DSP (Digital Signal Processor) [162], consuming about half the power of its synchronous counterpart.

Disadvantages

Although sufficient power reduction can be achieved by asynchronous design, asynchronous logic is not the main stream at this time, which means its lack of support by commercial CAD tools. EDA vendors have monitored academic developments of CAD tools for asynchronous design, but they have not yet included them into their products. Common layout libraries have been optimized for synchronous circuits. Although they are adequate for realizing asynchronous circuits [208], an optimized stand-cell library for asynchronous circuits can lead to further circuit area reduction. Furthermore, asynchronous circuits impose greater difficulty for the testability issues and have a higher cost overhead for design-for-testability.

2.2.6 Adiabatic switching and energy recovery

Adiabatic switching

The fundamental cause of CMOS dynamic power dissipation is the energy transportation in the circuit. When PMOS network is on, energy is injected into the circuit from the power supply. In a conventional CMOS circuit with load capacitance C and supply voltage V, the signal charge $Q = CV$ is drawn from the power supply and thus the injected energy

$E_{inj} = QV = CV^2$. Half of the injected energy ($\frac{1}{2}CV^2$) is stored in the load capacitor and the other half is dissipated. To reduce the power dissipation, designers may reduce the supply voltage; reduce the load capacitance or apply some combination of these techniques. In addition, there is a set of circuit design techniques targeting the minimal (asymptotically zero) energy dissipation during the charge transfer known as "adiabatic switching" or "adiabatic charging".

The word "adiabatic" comes from thermodynamics. It describes thermodynamic processes that exchange no heat with the environment. Here, the "process" is the transfer of electric charge between nodes in a circuit. The principle of adiabatic switching can be best explained by its comparison to a conventional dissipative switching. Figure 2.8(a) shows how energy is dissipated during a switching in a conventional CMOS circuit. A low to high transition at a node can be modeled as a charging of the load capacitor through a switch and the effective resistance. When the switch is turned on, C is charged up to V_{dd}. The current through the resistance decreases exponentially with the time elapsed. In an adiabatic circuit (Figure 2.8(b)), the transition is slowed down by using a time-varying voltage source instead of a static voltage supply. By spreading the transfer of charge to the capacitor over time, the current is greatly reduced. The overall energy dissipation is reduced to $\frac{RC}{T}CV_{dd}^2$ if the current flow is maintained constant ($E_{dis} = PT = I^2RT = (\frac{CV_{dd}}{T})^2RT = \frac{RC}{T}CV_{dd}^2$), where T is the total switching time, V_{dd} is the voltage increase at the node. Ideally, with T approaching infinity, the energy dissipation for a switching will approach zero.

Fully adiabatic circuit

Adiabatic switching has been incorporated into "reversible" computation techniques in order to achieve a *fully-adiabatic* [35] (asymptotically zero energy dissipation) circuit. Three

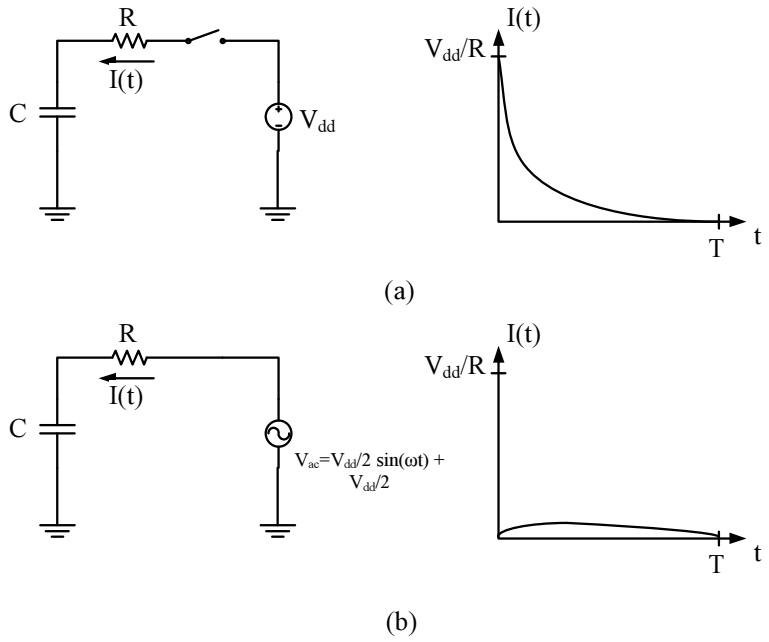


Figure 2.8: Modeling of the charging process in different circuits: (a) conventional CMOS, (b) adiabatic circuit.

decades ago, theoretical physicists found out that the energy requirement for a circuit can be potentially zero if computations can be implemented without loss of information [18]. Since then, a large body of work has been developed [19, 20, 136]. Ideally, by using reversible logic to avoid the destruction of information and by increasing the charging time T infinitely, the energy dissipation of a circuit can be made to approach zero. Athas et al. [9] showed the possibilities to assemble a fully adiabatic pipeline (Figure 2.9(b)) by constructing all of the logic stages (Figure 2.9(a)) and restricting the function blocks to be invertible only. We can see from Figure 2.9, the basic idea is to create the mirror image of a circuit element that computes the inverse of the original. For each stage, the computation result from the circuit element is passed on to its mirror image, where the inverse is computed. During the computation in the main circuit, charge is transferred to its end. It will flow back to the pulse power/clock source during the discharging phase through the mirror circuit of the function

in the next stage. Some other early work pursued reversible designs are SCRL (Split-level Charge Recovery Logic) [231], RERL (Reversible Energy Recovery Logic) [122], and the Pendulum reversible architecture [212]. Results in [122] show the energy consumption (per operation) of the RERL circuit (CLA and CPG) was reduced (about 6 times less than its static CMOS counterpart) for an operating frequency range between 50 and 70 kHz and a 5V supply voltage. Besides, a large portion of the energy is consumed by the CPG (clocked power generator), which was about five to ten times larger than that of the RERL CLA depending on the operating frequency.

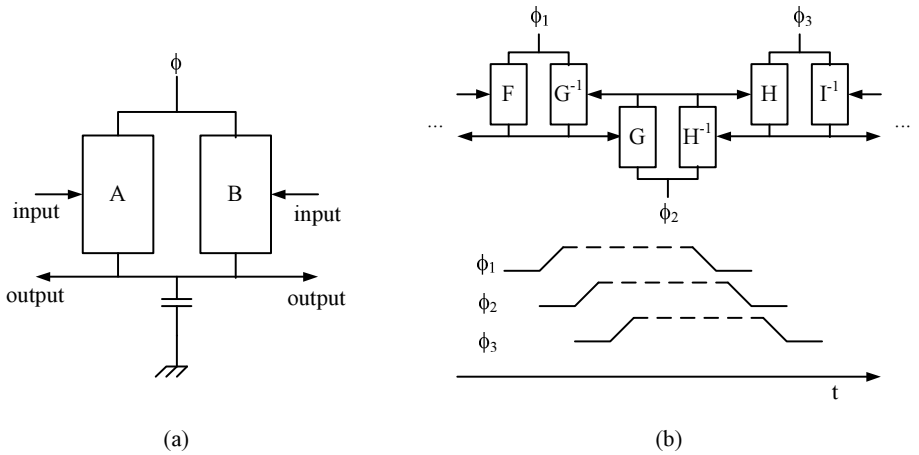


Figure 2.9: Conceptual adiabatic pipeline using invertible function: (a) the adiabatic gate in the pipeline, (b) a segment of pipelined adiabatic gates. In (a), the load capacitance may be charged through one functional network, A, and discharged through another, B. The input to the first network, A, must be valid during the charging phase. For simplicity, multiple switch networks needed for dual-rail signaling are not shown in (b). The corresponding pulse power/clock signal denoted as ϕ are also shown. One stage must be completely energized before the next stage commences.

Partially adiabatic circuits

An obvious shortcoming of the reversible design is that the circuit complexity is doubled, not counting the cost associated with the restriction of using a reversible function. The latter cost can be prohibitively high [8] and limits the usage of fully-adiabatic solution

only to very small circuits with simple functionality and cases where very slow switching is acceptable. Without the use of reversible logic, some researchers have proposed *partially-adiabatic* circuits with non-reversible design and non-zero asymptotic dissipation. Because some of the energy representing information in the circuits (in the form of charges stored in nodes) is recovered instead of being dissipated, "charge recycling" or "energy recovery" is used in such circuits. Early works focused primarily on adiabatic switching and resulted in a number of high-performance dynamic logic families [55, 57, 71, 105, 113, 127, 138, 153].

In [57], ADL (Adiabatic Dynamic Logic) was suggested. With a highly efficient clock supply circuit, an inverter chain using ADL achieves an average factor of 15 in power reduction over the conventional CMOS in the 1-100 MHz frequency range. ECRL (Efficient Charge Recovery Logic) was proposed in [138], which has a CVSL (cascode voltage switch logic) structure and requires 4-phase clocking for the efficient energy recovery. The ECRL inverter chain shows 10-20 times power gain over conventional CMOS in the 500k-10MHz frequency range. The 16-bit CLA using ECRL shows 4-6 times power gain (at 10MHz frequency) over a conventional CMOS implementation. CAL (Clocked Adiabatic Logic) proposed in [127] is a dual rail logic that can operate in either adiabatic mode or non-adiabatic mode using an AC power-clock supply or a DC power supply. The measured energy consumption in the adiabatic mode is about 8% of that in the non-adiabatic mode (at 10MHz clock frequency). PAL (Pass-transistor Adiabatic Logic) proposed in [153] is a dual rail logic with relatively low gate complexity supplied by a single two-phase AC power clock. The circuit can operate with a clock frequency up to 160MHz. It achieved a 2x power efficiency improvement in the 10-100MHz range as compared to earlier adiabatic logic families [55, 113].

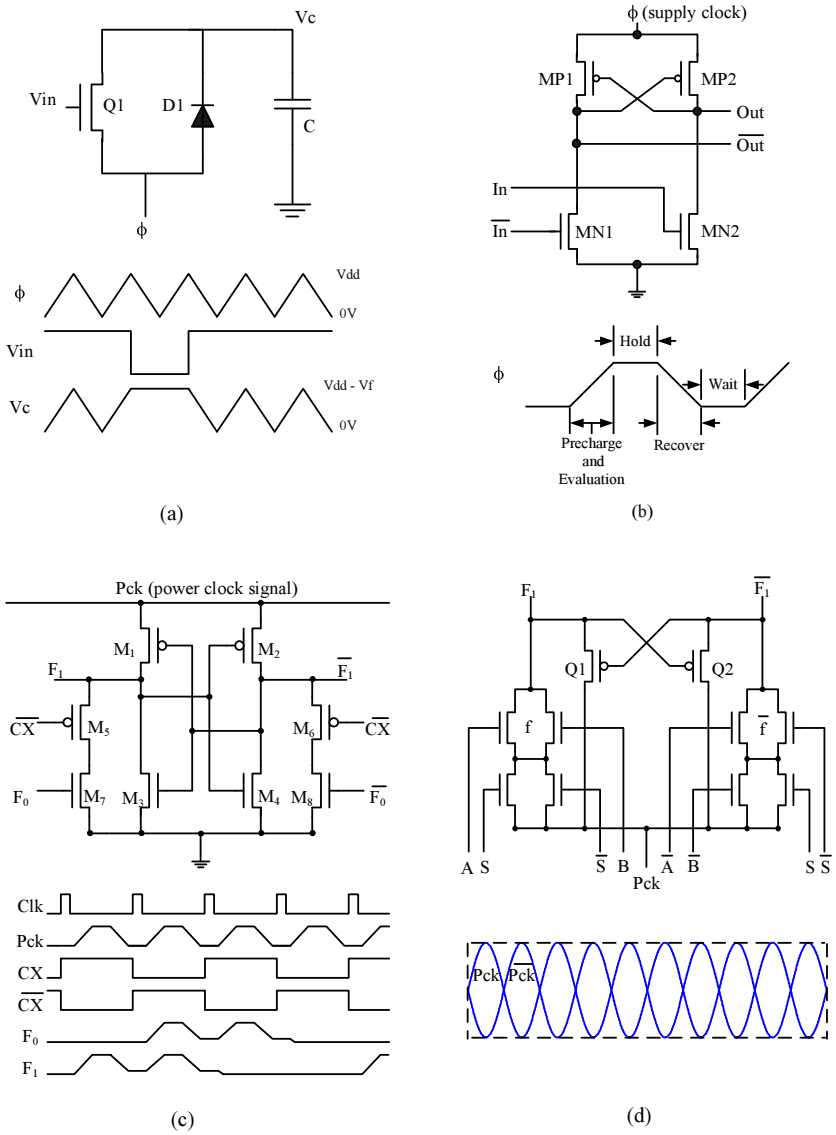


Figure 2.10: Example circuits for various adiabatic logic families: (a) ADL inverter and clocks, (b) ECRL inverter and the 4-phase clock, (c) CAL inverter and timing waveforms, (d) PAL multiplexer and power clock waveform. In (c), F_0 is the input signal and F_1 is the output signal; CX is the auxiliary clock. In (d), A, B, S are input signals and F_1 is the output signal.

Figure 2.10 shows some examples for the above logic families. For most of the other logic families mentioned above, a 4-5 times power improvement can be achieved at similar low operation speeds (sub-100MHz). Although some produced operational chips of considerable complexity [11, 12, 106], most of these designs were custom generated by hand.

Challenges and application

A major challenge in designing an energy recovery VLSI system is to design a highly efficient time-varying power source, the so-called power-clock generator. The key requirement for power-clock generators is the ability to transfer energy bidirectionally to and from the energy tank without much extra energy dissipation. This bidirectional charge transfer can be accomplished efficiently via a resonant power-clock waveform generated by an LC tank oscillator. Research on the design of such highly efficient resonant drivers has been reported [10, 57, 232], in the sub-100 MHz frequency range. Recently, an efficient energy-recovery clock generator was proposed. It can efficiently generate a sinusoid with a frequency higher than 100MHz [230]. For resonant circuits, the sinusoidal waveform has the highest energy recycling percentage.

Energy recovery has been applied to the static memory design [186, 203], showing considerable promise for reducing energy dissipation in SRAMs. In [203], an energy recovery SRAM is twice as power efficient as a conventional design at 200MHz. A more recent design [102, 233] has also shown 2x energy efficiency improvement over its conventional counterpart at 3V, 300MHz. In this approach, the energy recovery was applied to the clock distribution network and the word/bit lines of SRAMs.

A new energy recovery logic family resembling static CMOS is proposed in [230]. Although the power saving is far less than previous designs, it has a lower switching activity

than the dynamic energy recovery logic families and also possesses several positive characteristics of static CMOS. Some other applications for energy recovery techniques include a rotary traveling-wave oscillator that enables the recovery of charge from clock distribution lines operating at tens of GHz [217], and low-power drivers for LCD displays [6].

2.3 CMOS implementations and optimization

2.3.1 Dynamic power reduction

Transistor/gate sizing

Transistor sizing. Transistor sizing, which determines the sizes of transistors in a circuit, is an important step in the design process. In early days, when power consumption was not a major concern, improving the operational speed was the major objective of transistor sizing. Given the circuit topology, the delay of a combinational circuit can be controlled by varying the sizes of transistors. Here, the size of a transistor is measured in terms of its channel width/length ratio. In general, the delay of a gate can be reduced by increasing transistor widths from the minimum size. Hence, the transistor sizing problem often involves a trade-off between circuit area and delay. There has been much research on transistor sizing and the related optimization techniques [36, 47, 49, 67, 132, 178, 184, 190]. In general, heuristic algorithms [36, 49, 67, 184, 190] are relatively fast but they cannot guarantee the optimality of the solution. Non-linear programming approaches [132, 178] give exact optimal solution but suffer from the long run-time and convergence problem. Linear programming and piecewise linear approximation of nonlinear delay formulas have been proposed [22, 24] and are often fast and feasible for large circuits.

With the growing concern for low power dissipation, transistor-sizing techniques with consideration to power dissipation have emerged in the last decade. Since the major power dissipation in a logic circuit is the dynamic switching power, the power consumption is roughly proportional to its area (total capacitance).

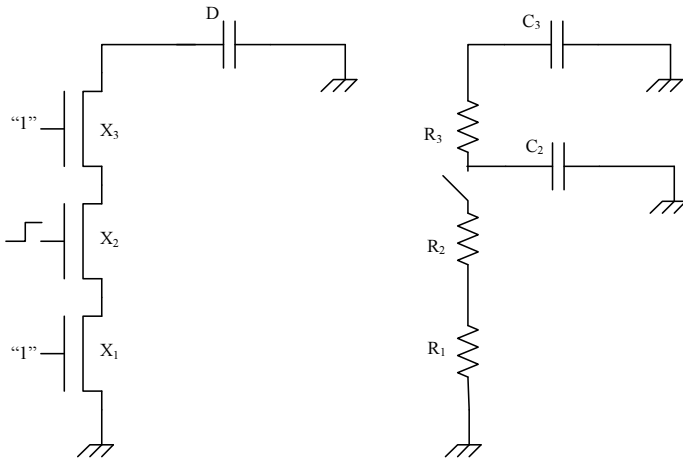


Figure 2.11: Timing model for TILOS: a pull down network is modeled as an equivalent RC network.

TILOS: an example. In the TILOS algorithm [67], the timing model for the circuit uses the Elmore delay formula [175]. The circuit is modeled as an RC network. As shown in Figure 2.11, each transistor is modeled as a perfect switch in series with a linear resistor. The gate, source, and drain capacitances are proportional to transistor size X, and transistor resistance is inversely proportional to X. Using the Elmore delay formula, the delay of the pull down network in Figure 2.11 is then

$$\begin{aligned}
&(R_1 + R_2)C_2 + (R_1 + R_2 + R_3)C_3 \\
&= (A/X_1 + A/X_2) * (B * X_2 + B * X_3 + C) + \\
&(A/X_1 + A/X_2 + A/X_3) * (B * X_3 + D)
\end{aligned} \tag{2.1}$$

where X_1, X_2, X_3 are transistor sizes, and A and B are technology parameters for transistor resistance and source/drain capacitance, respectively. C and D are wire capacitances. The path delay through an entire chain of logic gates can then be expressed as a function of transistor sizes,

$$\sum_{i,j=1}^{N} a_{ij}\frac{x_i}{x_j}+\sum_{i=1}^{N} b_i\frac{b_i}{x_i} \tag{2.2}$$

where a_{ij} and b_i are non-negative constants that depend on the circuit topology.

We see from the above, that the decreasing of the transistor size leads to a smaller load capacitance (and a lower dynamic power dissipation). However, the decreasing transistor size also leads to a lower drivability of the transistor (larger resistance). This means that the overall delay (modeled by $\tau = RC$ time constant) may not necessarily increase, which leaves the room for optimizations. Equation 2.2 belongs to a special class known as posynomials. A posynomial program requires the minimization of one posynomial while simultaneously satisfying a collection of upper bound constraints on other posynomials. Therefore, TILOS minimize the sum of transistor sizes (by adjusting each transistor size) under delay constraints. The optimization technique in TILOS is an iterative approach that starts with minimum-size transistors and then sizes them, iteratively. The technique is extremely simple to implement and has run-time behavior proportional to the size of the circuit. In one benchmark comparison with a mathematical programming technique [184], TILOS was found to converge to within 4.7% of the optimum in 8 out of 10 cases. TILOS solutions used 34% and 38% more power than the optimum solutions in the other two cases. TILOS can give a typical power reduction of 40% to 50% while maintaining the delay performance [66].

Other work. Borah et al. [26] have presented a direct approach to transistor sizing for minimizing power consumption of a CMOS circuit under delay constraints. They show that the power consumption is a convex function of the active area of the circuit instead of being proportional to it. Their analytical model for power dissipation includes both capacitive power and the short circuit power. A fast TILOS like heuristic algorithm is used to find the optimal transistor sizes. Experimental results show that further power reduction can be achieved if the objective of the algorithm is minimization of power instead of area. In [160], the transistor sizing problem for minimum power-delay product in nanoscale CMOS was formulated as a posynomial geometric program.

Yamada et al. [223] proposed a method to realize low-power dissipation by combining transistor sizing and transistor layout. When applied to a circuit with 10,000 transistors, the optimizer reduced the average transistor sizes to 1/8 of the original size while maintaining the same delay. The power dissipation was reduced to half when wiring capacitances were dominant. Hashimoto et al. [79] proposed a transistor sizing method that sizes MOSFETs inside a cell to eliminate redundancy in a cell-based circuit. Their method reduces power dissipation of an already routed circuits while preserving the interconnect geometry. The power dissipation is reduced by 77% maximum and 65% on average without increasing the delay.

In addition to switching power reduction, a method to reduce the short-circuit power was proposed in [25]. The idea is to size up the transistor that has a large fan-out. So the current drive is improved and the elapsed time for short-circuit current at fan-out gates is reduced. As will be discussed, much work has been done using transistor/gate sizing to reduce glitch power [23, 80, 104, 179, 219, 220].

Gate sizing. A related problem to transistor sizing is called gate sizing, where a logic gate in the circuit is modeled as an equivalent inverter and the sizing is carried out on this modified circuit with equivalent inverters in place of more complex gates [60]. This is an easier problem compared to the general transistor sizing problem because optimization is done for a smaller number of size parameters. Normally, techniques used in transistor sizing can also be applied to gate sizing [22, 190].

Glitch reduction

In conventional CMOS circuits, the spurious transitions at gate outputs due to the differential path delay are called *glitches* or *hazards*. As shown in Figure 2.12, hazards are due to the differing delays of logic blocks. The arrival times of signals at the inputs of a gate could be quite different, which lead to multiple transitions at the output before it settles to the correct logic value. As mentioned before, an 8-bit ripple-carry adder with a uniformly distributed set of random input patterns will typically consume an extra 30% in energy [34]. For a 16x16 bit multiplier with a logic depth of 30, hazards are found to consume as much as 67% of the total power ([168], p.45).

The elimination of hazards has been widely discussed in recent books [33, 168, 174]. The principal idea is to find delay assignment for all gates in the circuit to reduce the differential path delays at gate inputs with respect to the inertial delays. Published techniques of hazard elimination include, *balanced delay*, *hazard filtering*, *transistor sizing*, *gate sizing*, and *linear programming methods*.

Balanced delay. In the balanced delay (path balancing) method [14, 33], delays of all paths incident on a gate are equalized. When a signal fans out, its delay affects several

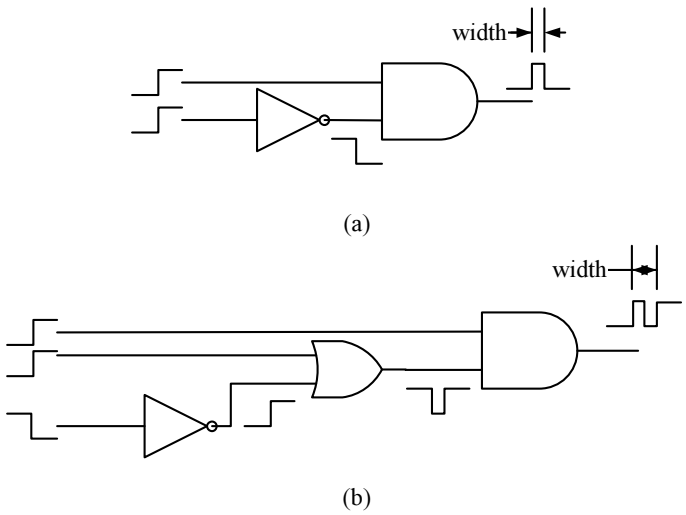


Figure 2.12: Hazard generation in logic circuits: (a) static hazard, (b) dynamic hazard.

paths and balancing requires the insertion of *delay buffers* on selected fan out branches. The advantage of this method is that delays are added only to the fast paths and the critical path delay of the circuit is not affected. In [104], both gate sizing and buffer insertion are adopted to achieve balanced paths. Experimental results show 61.5% glitch reduction and 30.4% power reduction without increasing the critical path delay.

Hazard filtering. In the hazard filtering method [3], it is assumed that if the width of a pulse is less than the inertial delay of the gate, the pulse will be suppressed or filtered out by the gate. This is known as the "filter effect" of a gate. Therefore, by adjusting the inertial delay to be greater than the differential path delay of the arriving inputs at the gate, glitches can be eliminated. Obviously, this method may increase the overall delay of the circuit. Figure 2.13 shows the difference between the hazard filtering and the balanced path methods. In [4], hazard filtering is applied to the circuit level design using a linear programming method (discussed later in this section) to find the optimal inertial delay for

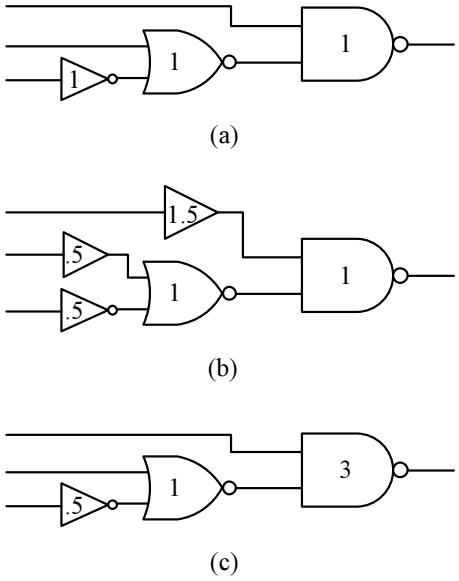


Figure 2.13: Examples for the balanced path and the hazard filtering method: (a) original circuit, (b) the balanced delay method, (c) the hazard filtering method. The number in each gate/buffer denotes its inertial delay.

each gate. Results show that a 4-bit ALU consumes only 53% peak and 73% average power after the optimization when the overall delay is allowed to increase.

Transistor sizing. The objective for transistor sizing is not only to find the delay assignments to eliminate glitches but also to fix the transistor sizes that would realize those delay assignments [25, 67, 179, 190, 219, 220]. In the recent work by Wroblewski et al. [220], balanced delay was considered in transistor sizing together with minimization of total capacitance and short-circuit power consumption. The solution is formulated as a multi-objective optimization, where the path delay difference and power consumption are the design objectives. Experimental results show that the power reductions for 4x4 and 16x16 multipliers are 32% and 45.6%, with 15% and 31% area increases, respectively. The advantage of this method is that it does *not* add buffers. However, it suffers from increased nonlinearity of the

delay model. It solves the problem in an unnecessarily large dimensional space by treating all transistors as parameters. Therefore, the numerical convergence is often a problem and a global minimization is not guaranteed [184].

Gate sizing. A technique similar to transistor sizing is gate sizing where sizes of gate are changed. It models a logic gate in a circuit as an equivalent inverter and carries out sizing optimization on the modeled circuit with equivalent inverters in place of the real gates [22, 23, 60]. Therefore, the number of parameters to be evaluated is far less than in the case of transistor sizing, which make it an easier problem than the transistor sizing problem. The gate sizes are allowed to vary in a continuous manner between a minimum and a maximum size. Similar to transistor sizing, the gate sizing techniques also suffer from the non-linearity problem of the delay model. Because mathematical solver needs the parameters (gate sizes in this case) to be continuously differentiable, a piece-wise linear simulator [22], or a non-linear programming solver [23] may be used to solve the problem. However, the complexity of these techniques limits the maximum size of circuit that can be analyzed and the optimality of the solution. [169].

Linear programming (LP). Linear programming techniques have been utilized to derive the delay assignments in a circuit [4, 170, 171]. A linear program determines a set of variables such that an objective is minimized under given constraints [68]. Circuit topology is formulated as a linear program and the delays of gate are treated as variables. The program returns the delay assignments for the given constraints and optimization objectives. In [4], linear programming has been incorporated with hazard filtering to determined the delay assignment for each gate. A single inertial delay is associated with each gate. It has

been shown that insertion of delay buffers is necessary if the objective is to eliminate all glitches and also control the overall delay. To enforce the overall delay constraint, the path enumeration method was used, which lists all possible paths from PI (primary input) to PO (primary output) and adds overall delay for each of them.

Later in [170], an improved linear constraint set method was proposed to replace path enumeration, which reduced the complexity of the constraint set from exponential to linear in the circuit size. Two new variables are introduced per gate in LP, i.e., earliest and latest arrival times of a signal. These two variables define the timing window in which signal can change at the output of a gate. Experimental results show 62% in average power and 66% in peak power reduction for a large ISCAS'85 benchmark circuit (c7552) without increase of overall delay.

To further eliminate the buffers inserted into the circuit and reduce the power consumption, Raja et al. [171] proposed a technique of designing gates with different input-output delays along different IO path through the gate. Thus, the gate consists of an inertial delay for the output and a set of delays for the input. Figure 2.14 shows an example of the delay model. In reality, there is an upper bound on the realizable delay difference along different IO path. Thus, a feasibility parameter is defined. Linear programming technique is used to find all the optimal delay assignments under the constraint of feasibility parameter and overall delays. Experimental results show up to an additional 24% power saving compared to the previous methods [170].

2.3.2 Leakage power reduction

Leakage is a major concern because it contributes to idle-power. It affects battery life even if the circuit is completely idle. As the minimum feature size shrinks, voltage

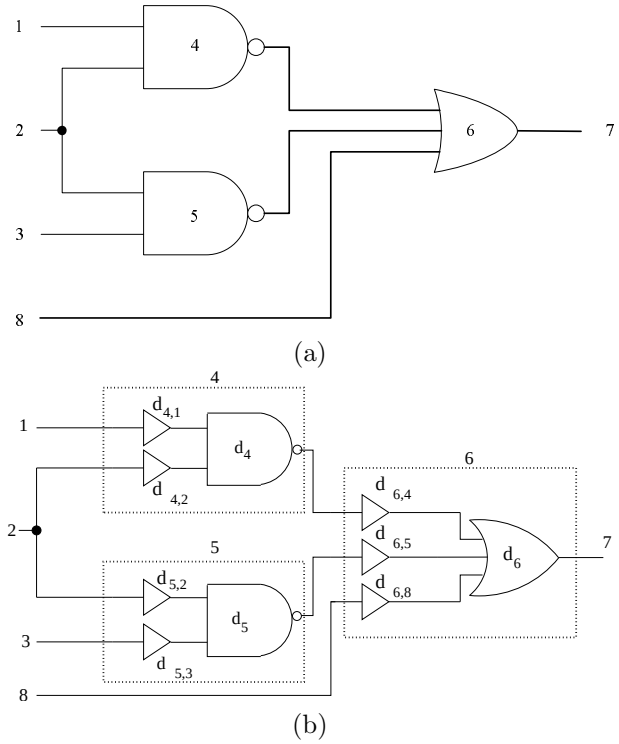


Figure 2.14: The variable input delay model in [171]: (a) the original circuit, (b) the delay model.

scaling requires the corresponding reduction in threshold voltage to avoid the exponential increase in delay. However, as threshold voltage decreases, the leakage current increases exponentially with each technology generation. From the projection analysis in [52], leakage current increases 7.5x per technology generation. Considering 30% supply voltage reduction per generation, the leakage power increases 5x per generation. At the same time, dynamic power increases much slower than the leakage power for constant die size. As technologies continue to scale, leakage power has become more problematic. To reduce the leakage, numerous techniques have been proposed, which can be classified into four categories: input-vector control, body-bias control, multiple threshold (dual-Vt), and power-supply gating. We will give a brief introduction to each of them.

Input-vector control

Input-vector control utilizes the "stacking effect" which describes the influence of an input pattern on the circuit leakage behavior [229]. Large reduction in leakage current can be achieved by simultaneously turning off more than one transistor in NMOS or PMOS "stacks" (i.e., series-connected devices) between supply and ground. During the standby mode, the input vector is selected to maximize the number of NMOS or PMOS stacks with more than one "off" device. In a study, circuits were simulated using sub-1V, 0.1μm technology. For a two-input NAND gate, it is demonstrated that the leakage current through a 2-transistor stack is approximately one order of magnitude smaller than the leakage of a single transistor. The implementation of a 32-bits CMOS adder in [229] shows up to 2x leakage reduction. To compensate the energy dissipated when entering and existing the standby mode (change of input vector), the adder must stay in standby mode for at least $5\mu s$ (minimum idle time).

Another related technique is called "forced stacking" [92], where an extra series-connected transistor in the pull-down path of a gate is inserted and turned off during the "stand by" mode. It offers a leakage reduction from 35% to as much as 90% relative to an unmodified circuit when a minimum leakage vector is applied.

Body-bias control

The body-bias control technique has many variations. However, they all have the same idea of adjusting the threshold voltage via body biasing. The threshold voltage of a short-channel NMOSFET transistor in the BSIM model [107] is given by,

$$V_{th} = V_{th0} + \gamma(\sqrt{\Phi_s - V_{bs}} - \sqrt{\Phi_s}) - \theta_{DIBL} V_{dd} + \Delta V_{NW} \tag{2.3}$$

where V_{th0} is the zero threshold voltage, Φ_s, γ and θ_{DIBL} are constant for a given technology. V_{bs} is the voltage applied between the body and source of the transistor. ΔV_{NW} is a constant that models narrow width effects, and V_{dd} is the supply voltage. Equation 2.3 shows that the threshold voltage will be lower than zero biasing threshold voltage when V_{bs} is positive, and vise versa.

Variable threshold CMOS (VTCMOS) [115, 117, 183] has been proposed to reduce the leakage current during the standby mode using a reverse body bias. As shown in Figure 2.15, when "SLEEP" is high ("1") for the standby mode, SSB (Self Substrate Bias circuit) is activated and body voltage V_{BB} for PMOS and NMOS transistors are raised to 4.3V and dropped to -2.0V individually. This leads to a higher threshold voltage and reduces the leakage current by 4 orders of magnitude [115]. When "SLEEP" is low ("0"), the SSB is disabled and MOS switches, MP1 and MN1, are turned on. Body voltages are reset to V_{DDL} and GND, which lead to zero biasing. As a result, V_{th} is set to 0.3V for fast circuit operation.

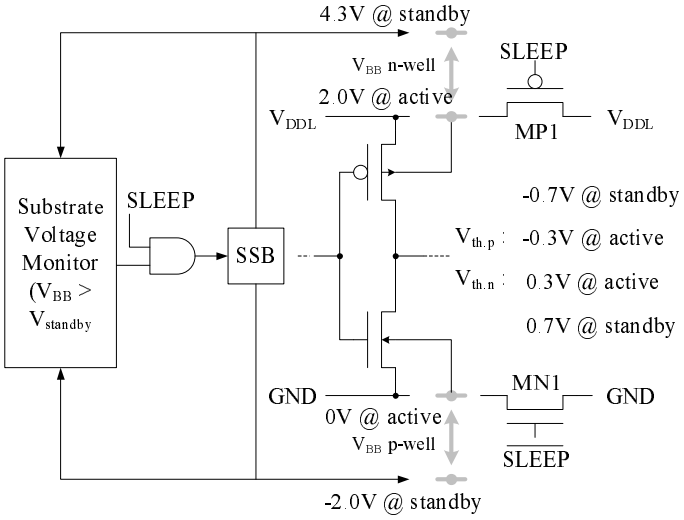


Figure 2.15: The illustration for the Variable Threshold CMOS Scheme.

Unlike VTCMOS, dynamic threshold CMOS (DTCMOS) [7] ties together the floating body and the gate of each transistor, resulting in a high (zero biasing) threshold voltage. From equation 2.3, we see that whenever the device is off, it has a high threshold voltage, which leads to a smaller leakage. Whenever the device is on, the lower threshold voltage allows higher current drive and speed.

A dual-Vt technique [38, 198, 215] provides two different threshold voltages (Vt) in the process. Depending on the path criticality in a circuit, transistors with high or low threshold voltages are used. Low-threshold transistors are fast and leaky, so they are assigned to speed critical paths. High threshold transistors are slow but leak less, and are assigned to non-critical paths. This technique does not change the threshold voltage of each transistor during the run time and thus has less overhead for control circuitry. Results in [38] show that the dual-Vt process brings about 2.5x improvement in energy-delay product over a single standard Vt process for an application with 98% idling factor. Results in [215] show the total active power can be reduced by around 50% and 20% at low and high-switching activities, respectively. For some circuits, both active and standby leakage power can be reduced by 80%. In [126], Lu et al. proposed a novel technique that uses integer linear programming (ILP) to minimize the leakage power in a dual-threshold CMOS circuit and simultaneously reduces the glitch power using the smallest number of delay elements to balance path delays. The constraint set size for the ILP model is linear in the circuit size. Experimental results show 96%, 40% and 70% reduction of leakage, dynamic and total power, respectively, for the benchmark circuit C7552 implemented in the $70nm$ BPTM CMOS technology.

Power-supply gating

The final approach we will discuss here is power-supply gating. The basic idea is to shut down the power supply so that the leakage power of idle units is almost zero. This is done by inserting "sleep transistor" to cut the path from the power supply to the unit [140] or by controlling of power supply regulators [59]. As proposed in [140], high-Vt devices are inserted in series with low Vt circuitry. These are called sleep transistors. In this way, virtual supply and/or ground are created with voltage level very close to the real V_{dd} and GND. In the standby mode, the sleep transistors are turned off by sleep control signals. The path from power supply and/or ground to the unit is cut off. Figure 2.16 shows an example of this approach. In [59], PLL (Phase Locked Loops) based power supply regulator [213] was used to turn off the power supply in the standby mode. Simulation results showed an almost complete elimination of leakage.

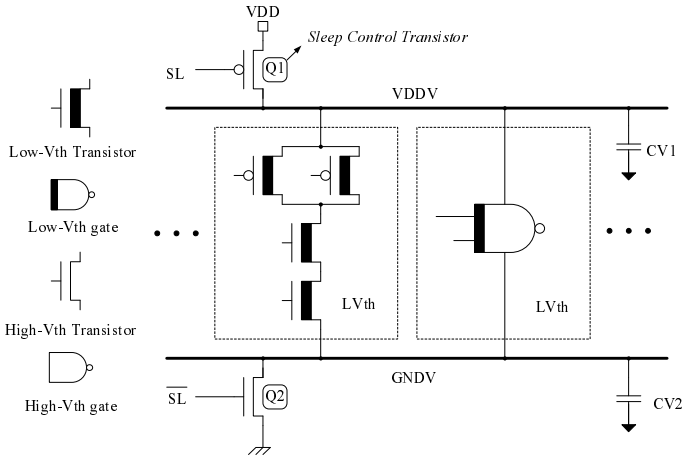


Figure 2.16: A example of using sleep transistors to gate supply power.

Efficiency comparison

The efficiency of various leakage reduction techniques has been compared in the literature [59, 98, 199]. In a recent study by Tsai et al. [199] the implications of technology scaling on leakage reduction techniques are studied. Three major leakage reduction techniques introduced above are evaluated and compared under $0.25\mu m$, $0.18\mu m$ and $0.07\mu m$ technologies. Both sleep transistor (called gated V_{dd}) approach and power supply regulator approach to gate the power supply are simulated. Simulation results suggest that input vector control and power supply gating will become more efficient when technology scales down. However, the effectiveness of body-bias control decreases as technology scales. A similar conclusion is made in [98]. However, even though the effectiveness decreases, the leakage reduction by body-bias control is still significant for $0.07\mu m$ ($> 50\%$ on average). The minimum idle time for all techniques decreases due to increasing ratio of leakage to total power.

Comparing these three techniques for the $0.07\mu m$ technology, the input-vector control has a leakage reduction of 6% to 76% (for different circuits), body-bias control has a leakage reduction of 40% to 85%, power-supply gating has leakage reduction of 88% to 98% and power-supply regulator approach has virtually 100% leakage reduction. Input-vector control has the largest area overhead (0.26% to 18.7%) and worst minimum idle time (0.17 to $110.8\mu s$). Gated V_{dd} has a smallest area overhead (0.34% to 2.5%) and minimum idle time (0.2 to $4.5ns$).

2.4 Summary

In this chapter, we give a survey of device/circuit level low power techniques. The specific low power techniques covered by our survey include CMOS, domino CMOS, PTL (pass transistor logic), self-timed logic, asynchronous design, adiabatic switching and energy recovery. CMOS optimization techniques are classified into dynamic power optimization techniques and leakage reduction techniques. For each technique, the basic idea is illustrated with examples. The effect on power reduction is discussed. The technique proposed in this dissertation falls in the category of glitch reduction techniques of dynamic power optimization for CMOS circuits.

Chapter 3

Power Estimation Techniques

Power estimation refers to the techniques that can estimate or predict the average power and maximum power for a given circuit. Power estimation is critical to any design because the power consumption must meet the specification during the design phase. Otherwise, a costly redesign process will be inevitable. In this chapter, we give an introduction to various power estimation techniques. Both simulation-based approaches and non-simulation approaches are illustrated.

3.1 Simulation-based approaches

Circuit simulation based techniques simulate the circuit with a representative set of input vectors and calculate the average power consumption. The advantage of this approach is that it is accurate and is applicable to any circuit regardless of technology, design, style, functionality, architecture, etc. However, it has two major drawbacks. First, it requires large memory and execution time and is not suitable for large circuits. Second, it is known that the power estimation by this approach is strongly input pattern dependent [96, 222]. The second problem is serious because the input patterns may not be known to the designer when the power of a functional block is estimated. The input patterns are determined by the system environment that the functional block is embedded in. To ensure the accuracy of the power estimation, a large number of input patterns has to be simulated, which makes it time consuming and computation intensive.

3.1.1 Circuit-level simulation

SPICE (Simulation Program with IC Emphasis) [141] is the de facto power analysis tool at the *circuit level*. SPICE solves a large matrix of nodal current equations derived from the Krichoff's Current Law (KCL). The basic components of SPICE are the basic circuit elements such as resistors, capacitors, inductors, current sources and voltage sources. More complex device models such as diodes and transistors are constructed from the basic components. Basic circuit parameters, e.g., voltage, current, charge, etc., are reported by SPICE simulation with high accuracy. The power dissipation can then be derived from those parameters. The strongest advantage of SPICE is its accuracy. With a correct device model, SPICE simulation can reach accuracy within a few percent of physical measurement. However, the intensive computation requirement limits the application of SPICE for large circuits.

One way to speed up computation in SPICE is to express the transistor model in a tabular form stored in the database. Instead of evaluating equations, a simple table lookup can find out the current value corresponding to the input voltage. PowerMill [54] is a such kind of power simulator and analyzer, which also applies an event-driven timing simulation algorithm. An event is registered when a significant change in node voltage occurs. If the event driven approach fails it rolls back to circuit analysis method. The tabular transistor model introduces inaccuracies but significantly improves the speed of analysis, which is about two orders of magnitude faster than SPICE.

Switch level simulation views a transistor as a two-state switch with a resistor. The switch is turned on when its gate voltage is above the threshold voltage. Under this model, simulation can be performed using an approximate RC calculation that is more efficient

than the transistor level analysis. Switch-level simulation tools have been reported [28, 81]. Standard switch-level simulators (such as IRSIM [177]) can be easily modified to report the switched capacitance (and thus dynamic power dissipation) during a simulation run. Short-circuit power can be accounted for by observing the time in which the switches form a power-to-ground path. Obviously, switch-level simulation is less accurate than the circuit-level simulation but offers faster speed.

3.1.2 Gate-level simulation

Gate-level timing analysis is a matured technique. The component abstractions at this level are logic elements, such as, NAND gates, latches, flip-flops, and nets. The most popular gate-level analysis is based on the event-driven logic simulation. Events are zero-one logic switchings of nets in a circuit at various given times. As one switching event occurs at the input of a logic gate, it might trigger another event at the output of the gate after a time delay. Power consumption at each node can be calculated from the switching activity and capacitance of the node. Internal power (power consumption by nodes inside a logic cell) and static power (leakage power of the gate) are calculated based on power macromodels (power dissipation related to input events and static state). Verilog-XL logic simulator from Cadence Corp. is a Verilog-based gate-level simulation program using gate-level timing analysis for power estimation. The accuracy of the estimation depends on the accuracy of the macromodels built for the gates in the ASIC library, the glitch-filtering scheme used, and the accuracy of physical capacitances provided at the gate level. The speed is 3-4 orders of magnitude faster than SPICE.

Monte Carlo simulation has been proposed [31, 221] to estimate average power statistically. The basic idea is to simulate a circuit with increasing number of vectors until the

average power estimate converges. It consists of applying randomly generated patterns at the primary inputs and monitoring the energy dissipated per clock cycle using a simulator. If the successive input patterns are independently generated, a number N of such measurements is called a random sample whose average approaches the true average energy in a clock period (average power) for large N. To stop the simulation when it is close to the average power, a stopping criterion is needed. It was found experimentally [31] that the energy consumed by a circuit over a time period T has a distribution very close to *normal*. This allows the derivation of stopping criterion from the sample average and sample standard deviation, given a user specifies the required confidence level and percentage error. In other words, one can assure with the specified confidence level that the measured average power is within the user specified error range with respect to the actual average power.

3.1.3 RTL simulation

Register Transfer Level (RTL) abstraction contains basic building modules like registers, adders, multiplier, busses, multiplexers, memories, state machines, etc. A power macromodel for each module is normally built by simulating it under pseudo-random data and fitting a multi-variable regression curve (i.e., power macro-modeling equation) to the power dissipation result using a least mean square error fit [16]. The macromodel may be parameterized in terms of input bit width, the internal organization/architecture of the component (capacitive components and bit line activities), and supply voltage level [78, 119, 125, 166]. Reader can refer to a recent survey by Pedram [37] for more details about RT-level power macro-modeling.

After the power macro-modeling for RT-level components, power estimations at RT-level can be implemented in the form of a power-cosimulator for standard RT-level simulators. The power-cosimulator is responsible for collecting input data statistics for all RT-level modules from the output of the RTL simulator and producing the power value. Since evaluating the macromodel equation at every clock cycle during simulation may have a high overhead (of data collection and macromodel evaluation), Hsieh et al. use simple random sampling to select a sample set and calculate the macromodel equation for the vector pairs in the sample set [83].

3.1.4 High level analysis

Most of the high level power prediction tools use profiling and simulation techniques to address data dependencies. Important statistics include the number of operations of a given type, the number of bus, register and memory accesses and the number of I/O operations executed within a given period [32, 114] . Instruction level simulation or behavioral simulators can be adapted to produce this information.

In [84], Hsieh et al. presented an approach called profile-driven program synthesis for power estimation of high-performance CPUs. Instead of using a macro-modeling equation to model the energy dissipation of a microprocessor, they used a synthesized program to exercise the microprocessor in such a way that the resulting instruction trace behaves similar (in terms of performance and power dissipation) to the original trace. However, the new instruction trace is much shorter than the original one and hence can be simulated on an RT level description of the target microprocessor to provide the power dissipation quickly.

3.2 Non-simulation approach

3.2.1 Behavior level analysis

At the behavior level, not much information is available about the gate-level structure. Hence, abstract notions of physical capacitance and switching activity are used to predict power dissipation. These techniques can be classified into three broad categories: information theory based, complexity based, and synthesis based approaches.

Information theory based approaches

Information theory based approach [129, 146] depends on information theoretic measures of activity (i.e., entropy) to estimate power dissipation. Entropy characterizes the randomness of a sequence of vector and hence is related to the switching activity. It is shown in [129] that, under temporal independence assumption, switching activity of a bit is upper bounded by 1/2 of its entropy. The power dissipation in the circuit can be expressed as $Power = \frac{1}{2}V^2 f C_{tot} E_{avg}$, where C_{tot} is the total capacitance of the logic module and E_{avg} is the average of line activities, which is in turn approximated by 1/2 of the average entropy h_{avg}. The average line entropy h_{avg} is calculated by a closed-form expression parameterized by average bit-level entropies of circuit inputs/outputs (and number of inputs, outputs). Average input entropy can be derived from input sequences. Average output entropy is derived either by using an *effective information scaling factor* and number of logic level in the circuit if gate-level structure is given; or by a compositional technique based on pre-characterization of library modules in terms of their entropy transmission coefficient if only functional/data-flow information is given.

In [146], word-level entropy is used instead of bit-level entropy. A similar closed-form expression for h_{avg} is proposed using sectional (word-level) input/output entropy. The sectional entropies of circuit inputs and outputs may be obtained by monitoring input output signal values during a high-level simulation of the circuit. In practice, they are approximated as the summation of individual bit-level entropies. The total module capacitance can be calculated by summing up the entire gate loading and wire capacitance if gate-level structure is given. Otherwise, C_{tot} is estimated by a quick mapping (e.g., mapping onto universal gates) or by information theoretic models that relate the total capacitance to input and output entropies [39, 65].

Complexity-based approaches

Complexity-based models relate the circuit power to the circuit complexity. Most of the proposed complexity-based models rely on the assumption that circuit complexity can be represented by the number of "equivalent gates". Muller-Glaser et al. proposed a *chip estimation system* [139] that computes the average power of a logic module as $Power = fN(Energy_{gate} + 0.5V^2C_{load})E_{gate}$. Here, f is the clock frequency, N is the equivalent gate count for this module, $Energy_{gate}$ is the average internal energy dissipation for an equivalent gate, C_{load} is estimated capacitance based on the average fanout in the circuit and the wire load model, and E_{gate} is average output activity per clock cycle for an equivalent gate. E_{gate} is dependent on the functionality of the module. These data are pre-calculated and stored in a library and are independent of the implementation style and the circuit environment.

In [148], Nemani et al. presented a high-level estimation model for predicting the area of an optimized single-output Boolean function. The model is based on the assumption that the area complexity of a Boolean function is related to the distributions of the sizes of the

on-set and off-set of the function. Area measure is used for total capacitance estimation and hence high-level power estimation. This work has been extended to area estimation of multiple output functions [147].

Complexity-based power prediction for controller circuitry was proposed by Landman and Rabaey [120]. Based on the knowledge of its target implementation style (i.e., pre-charged pseudo-NMOS or dynamic PLA), the number of inputs, outputs, input/output activities, etc., this techniques can give quick power estimation. The accuracy of the estimates depends on the empirical parameters (regression coefficients), which are derived by curve-fitting and least-square fit error analysis on low-level simulation of previous design.

Synthesis-based approaches

Synthesis-based models assume an RT-level template and produce estimates based on that assumption. It requires the development of a quick synthesis capability that makes the relevant behavioral choices. Important behavior choices include type of I/O, memory organization, pipeline issues, synchronization scheme, bus architecture, and controller design. After the RT-level structure is obtained, power consumption can be estimated by either simulation or static analysis of the circuit structure/functionality.

3.2.2 Gate-level probabilistic approach

Dynamic power has been the dominating component of the total power consumption. Since dynamic power can be estimated by the switching activity and capacitance at circuit nodes, one can estimate the power consumption of the circuit by deriving the switching activity using probabilistic measures. Several different approaches have been proposed that use probability to derive the power consumption at the gate level. These estimation methods

differ in various aspects, such as delay model, spatial and/or temporal correlation among signals, estimation for individual gate, etc. They also vary in the estimation complexity and speed. Most of them focused on combinational circuits.

Signal probability and transition probability

In [44], which is one of the early works using probabilistic approach for power estimation, zero (gate) delay model is used. Therefore, glitch power is not considered. In addition, spatial independence of signals is assumed. *Signal probability* (probability of a signal equals one, denoted as P_s) is propagated into the circuit from primary input using basic probability theory. For a two-input AND gate $y = AND(x_1, x_2)$, signal probability $P_s(y) = P_s(x_1)P_s(x_2)$ under the condition that inputs are independent. The *transition probability* (probability of signal switching, denoted as P_t) is calculated from signal probability under temporal independence assumption, i.e., transition probability $P_t(y) = 2P_s(y)(1 - P_s(y))$. To derive signal probability efficiently, OBDD (ordered binary decision diagram) based method has been proposed [29]. In this method, the signal probability at the output of a node is calculated by first building an OBDD corresponding to the global function of the node (i.e., function of the node in terms of the circuit inputs). Then a traversal of the OBDD is performed using equation: $P_s(y) = P(x_1)P(f_{x_1}) + P(\bar{x}_1)P(f_{\bar{x_1}})$, where f_{x_1} and $f_{\bar{x_1}}$ represents Boolean functions when x_1 equals one and zero, respectively. This OBDD-based method leads to an efficient calculation of signal probability.

In [63], pair-wise signal correlations are used in propagating signal probability. Higher order correlations are approximated by pair-wise correlations. In [130, 180], transition correlations are used to describe the spatial temporal correlation between two signals in consecutive clock periods. Signals in consecutive clock cycles are modeled with lag-one

Markov chain under a zero delay assumption. The transition probabilities can be computed exactly using the OBDD based approach in terms of circuit inputs. In [130], author also proposed a faster way of propagating transition probabilities without using global OBDD. The loss of accuracy is small with significant computation savings. This work has been extended in [131] to handle highly correlated input streams with two new concepts, conditional independence and isotropy of signals. Based on these, a sufficient condition for analyzing complex dependencies is given.

Probabilistic simulation

All above approaches make the zero delay assumption, which means that glitch power is not considered. In reality, power consumed by glitches is not negligible. In [197], transition probability was extended to the real delay case. Many other approaches are based on real delay model or differential delay to account for the glitch power. Probabilistic simulation (CREST) [30, 143, 144] models signal with a probability waveform. As shown in Figure 3.1, the probability waveform is a sequence of values indicating the probability that the signal is high for certain time intervals, and the probability that it makes high-to-low and/or low-to-high transitions at specific time point. The propagation algorithm is like event driven logic simulation with assignable delays and the only difference is that probabilistic simulation at each gate deals with the *probability* of making a transition rather than a definite occurrence of a transition. The spatial correlation is not considered in this approach. Later on, a tagged probabilistic simulation (TPS) [58, 201] was proposed which considers the spatial correlations among signals. As shown in Figure 3.1, the probability waveform of a signal in one clock period is divided into four different tagged waveforms based on the steady state signal transitions, i.e., 00,01,10,11. The correlations between steady-state signals

are used to approximate spatial correlations between the intermediate signal values. The transition correlations can be derived using methods in [130, 131] or by simulation. It is more efficient than trying to estimate the correlation between intermediate signal values while the estimation accuracy is improved when compared to the case without spatial correlations.

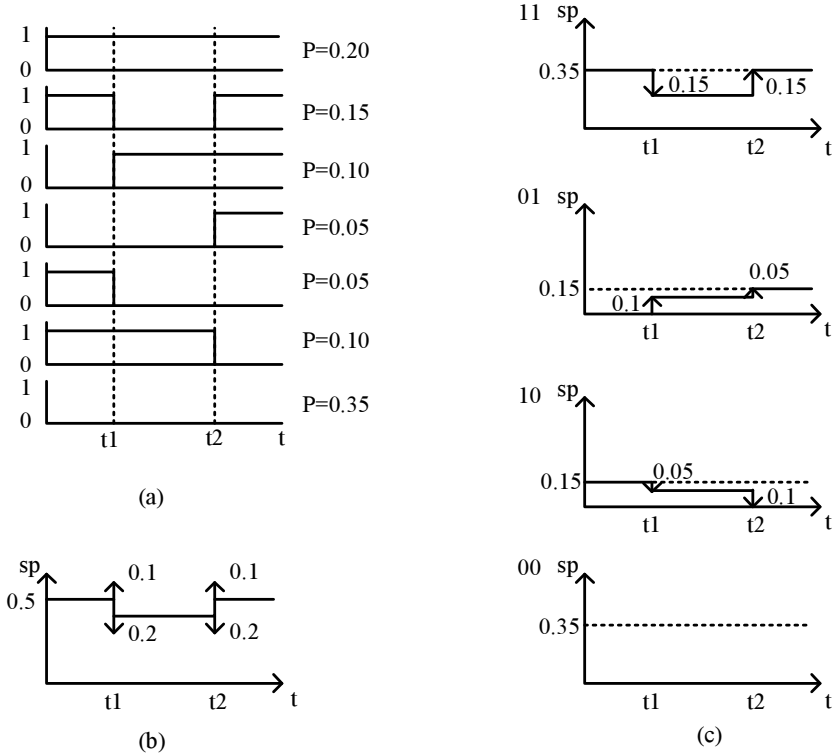


Figure 3.1: Illustration of probability waveforms: (a) logic waveforms with corresponding occurring probabilities, (b) corresponding probability waveform, (c) corresponding tagged probability waveform.

In [85, 86], Hu et al. proposed a new glitch filtering analysis using the dual-transition probability that captures the states of a node at two different time instances. Experiments show that probabilistic simulation and the TPS techniques, when enhanced by the dual-transition analysis, provide more consistent power estimation. Experimental results on

ISCAS'85 benchmarks show significant improvements in estimation accuracy as the average estimation error on total power consumption remains under 5%.

Transition density

Another widely used notion, *transition density*, was proposed by Najm [142] for power estimation. Transition density is the average transitions in one clock period at a node. A companion concept is equilibrium signal probability, which is the average signal probability over an infinitely long time. To propagate the transition density, the concept of Boolean difference is used. If y is a Boolean function that depends on x, the Boolean difference can be expressed as,

$$\frac{\partial y}{\partial x} \triangleq y|_{x=1} \oplus y|_{x=0}$$

where $\oplus$ denotes exclusive-or. It was shown in [142], that if inputs x_i to a Boolean module are spatially independent then the density of its output y is given by:

$$D(y) = \sum_{i=1}^{n} P(\frac{\partial y}{\partial x_i}) D(x_i)$$

where transition density is denoted as D. Propagation of transition density is based on the differential delay assumption, that is, no two transitions happen at the same time.

Other works

Although most work on probabilistic power estimation focuses on combinational circuits, some works have been done on sequential circuits [75, 137, 202]. Switching activity estimation is much more difficult for FSMs (finite state machines) because, first, the probability of the circuit state needs to be calculated; second, the present state line inputs of

FSMs have strong space and time correlations. The basic idea in [75] is to unroll the next state logic once, and then perform symbolic simulation on the resulting circuit. This method does not capture the spatial correlations among present state lines and makes a simplistic assumption that the state probabilities are uniform. The above work is improved upon in [202] and [137] which use the Chapman-Kolmogorov equations for discrete-time Markov chains to compute the exact state probabilities of the machine. The Chapman-Kolmogorov method requires the solution of a linear system of equations of size $2N$, where N is the number of flip-flops in the machine. This method is limited to circuits with a small number of flip-flops because it requires the explicit consideration of each state in the circuit.

3.3 Summary

In this chapter, various power estimation techniques for different levels of abstraction of circuits are discussed. Generally, power estimation at a lower level has a better accuracy than at the higher level. However, more details of the circuit and computation resources are required for the lower level estimation. Two major approaches in power estimation are simulation-based approach and the non-simulation methods. Depending on the accuracy requirement and available information of circuits, various power estimation methods can be adopted accordingly.

Chapter 4

Process Variations and Our First LP model

In this chapter, we present our first process-variation-resistant LP model. We briefly review the basics of linear programming and the previous LP model [170]. Then, we discuss the sources of process-variation and their effects on delay and power variations. Previous work related to process variations is reviewed and discussed. We build our statistical gate delay model assuming that delay variations have normal distributions and show that the effect of inter-die variations on power dissipation of a circuit is negligible. We prove that, in some cases, it is necessary to increase overall circuit delay to obtain a glitch-free design under process-variations. Our first process-variation-resistant LP model is then constructed based on a worst-case timing analysis.

4.1 Background

4.1.1 Basics of linear programming

Linear programming, also referred to as operations research, optimization theory, convex optimization theory, or linear optimization, is a method of maximizing (minimizing) a linear function over a convex polyhedron [216]. Linear programming is extensively used in economics and engineering. A linear program determines a set of variables such that an objective is minimized under given constraints [68]. The problem is expressed in the

following form.

$$
\begin{aligned}
&\texttt{minimize} && c_1x_1 + c_2x_2 + ... + c_nx_n \\
&\texttt{subject to} && a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n \leq b_1 \\
& && a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n \leq b_2 \\
& && ... \\
& && a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n \leq b_m \\
& && x_1, x_2, ...x_n \geq 0
\end{aligned}
\tag{4.1}
$$

where x_i ($i \in [1, n]$) are variables, and a_{ji}, b_j, and c_i ($i \in [1, n], j \in [1, m]$) are constants.

A linear program can be solved using the simplex method [50, 218] (1949) which runs along polytope edges of the visualization solid to find the best answer. In 1979, Khachian [101] found a $O(x^5)$ polynomial time algorithm. A much more efficient polynomial time algorithm was found by Karmarkar [97] (1984). This method goes through the middle of the solution space (the so-called interior point procedure) and then transforms and warps the space to quickly reach a solution point.

In our implementation, we use AMPL modeling language to construct and solve the linear program. AMPL is a comprehensive and powerful algebraic modeling language for linear and nonlinear optimization problems with discrete or continuous variables. Developed at Bell Laboratories, AMPL allows us to use common notations and familiar concepts to formulate optimization models and examine solutions while the computer manages the communication with an appropriate solver [68].

4.1.2 Previous LP approach for low power

Although we have already mentioned the previous LP models [170] in chapter 2, it is beneficial to review them here in greater details, so that we can see the limitation of that model.

Concept of timing window

Instead of having a single parameter for a gate as was done earlier [3, 4], Raja [170] introduced the concept of timing window which contains two variables indicating the earliest and latest signal arrival times at the output of a gate. For gate i with n inputs, variables t_i and T_i are defined as the minimum and maximum time instant at which an event can occur at the output of the gate after the occurrence of an event at PIs (primary inputs) of the circuit. Figure 4.1 shows the concept of this timing window.

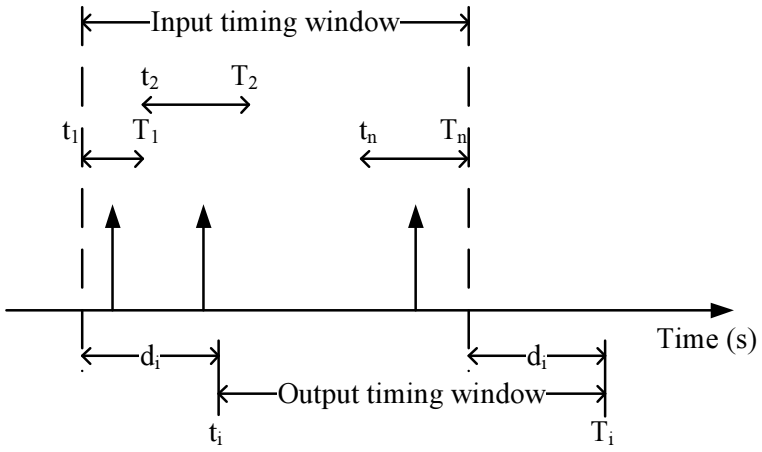


Figure 4.1: The illustration of the timing window at gate i.

It is well known [3] that if the input pulse width is less than the inertial delay of a gate, the pulse will be suppressed or filtered by the gate. This is referred to as the "filter effect" of the gate. Therefore, by adjusting the inertial delay to be greater than

the differential path delay of arriving inputs at the gate, glitches can be eliminated. This technique, known as *hazard/glitch filtering*, is adopted in the LP model in [170] with the help of above timing window. To obtain a design without increasing the overall circuit delay, *path balancing* (illustrated in Section 2.3.1) method is also adopted in [170], where path delays are balanced via the adjustment of gate delays and insertion of buffers.

Linear program

We illustrate the linear programming model using the example of the adder circuit shown in Figure 4.2. Buffers are inserted at PIs and at each fanout branch of a signal that has more than one fanout. The linear program is developed as follows.

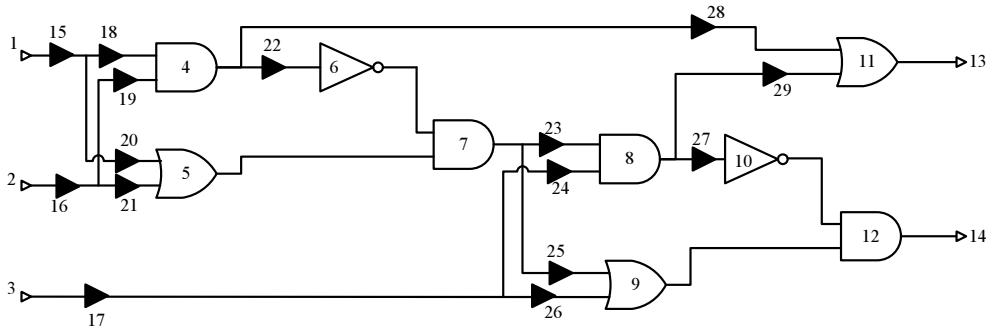


Figure 4.2: The 1-bit adder circuit. Black triangles represent buffers inserted. Gate number is marked for each gate and buffer.

Variables: Variables can be split into two categories, gate variables and buffer variables. The gate variables for each gate i are:

- T_i, the maximum time at which the output of gate i can produce an event after the occurrence of an event at PIs.

- t_i, the minimum time at which the output of gate i can produce an event after the occurrence of an event at PIs.
- d_i, the inertial delay of gate i, which is to be determined by the optimizer.

The buffer variables also have the same sets of parameters as gate variables. However, they are treated differently in the program.

Objective function: The injection of buffers into the circuit increases the area and power of the circuit and the objective would be to reduce the number of buffers. However, this is a non-linear objective. Hence, the objective function in [170] is to reduce the sum of the buffer delays, which is equally effective in achieving the goal.

Constrains: Initial constraints specify a lower bound on each variable. These are constraints $d_i \geq 1$ for each gate i, $d_i \geq 0$ for each buffer i, $T_i \geq 0$ for each gate and buffer i, and $t_i \geq 0$ for each gate and buffer i.

Gate constraints are different for inverters/buffers and multi-input-gates. For example, the buffer 19 in Figure 4.2 has a set of constraints:

$$\begin{aligned} T_{16} + d_{19} &= T_{19}; \\ t_{16} + d_{19} &= t_{19}; \end{aligned} \tag{4.2}$$

Considering the case of multi-input-gates, as for gate 7, the constraints are:

$$\begin{aligned} T_7 &\geq T_5 + d_7; \\ T_7 &\geq T_6 + d_7; \\ t_7 &\leq t_5 + d_7; \\ t_7 &\leq t_6 + d_7; \\ d_7 &> T_7 - t_7; \end{aligned} \tag{4.3}$$

The first four constraints ensure that the parameter T_7 settles at $Max(T_5, T_6)$ and t_7 would settle at $Min(t_5, t_6)$. The last condition ensures the hazard filtering condition.

To ensure that the delay balancing and hazard filtering do not slow down the circuit beyond the specific limit; there is an upper bound on the maximum delay at POs (primary outputs). This can be ensured by placing upper bounds on parameter T of all gates feeding the primary outputs of the circuit. Thus, there are additional constraints as:

$$\begin{aligned} T_{11} &\leq Maxdelay \\ T_{12} &\leq Maxdelay \end{aligned} \tag{4.4}$$

Observations

An assumption made in the above model is that each gate has a single fixed inertial delay. The LP solution derived from this model guarantees the elimination of glitches. However, since there is no consideration of variations of gate delays, a solution provided by the above model is very sensitive to the change of gate delays. Small changes of gate delays can corrupt glitch elimination conditions and result in the degradation of power dissipation.

In real circuits, the delay of a gate is not deterministic but is rather a random variable because of the existence of variations in manufacturing, operation temperature, supply voltage, etc. For the deep sub-micron technologies, the variations in device parameters are higher due to the increasingly difficult fabrication process. Thus, gate delays can deviate from their design values dramatically and result in a large increase of power dissipation from the expected value.

In this chapter, we introduce our first of two process-variation-resistant LP models that take the process (delay) variations of gates into account during the optimization. Our goal is to derive a robust solution that is less sensitive to the delay variations of gates. That is, the optimized circuit maintains low power dissipation even though gate delays could deviate.

Note that in [170], buffers inserted in a circuit consume additional power. To reduce the power introduced by the additional buffers, Raja has improved the work in [170] by designing a new type of gate that has differential input delays [169, 172]. Furthermore, in [204, 205], the author has shown that it is possible to replace all traditional buffers (two inverters in series) in a circuit with resistance type of buffers, which consume roughly zero additional power. In this dissertation, we assume that all the buffers inserted into the circuit are of resistance type and do not consume additional power.

4.2 Process and delay variation

4.2.1 Process variation

Process variations refer to the variations due to the semiconductor process, such as threshold voltage, oxide thickness, device length (L_{eff}), interconnect wire width, thickness,

etc. In general, process variations can be divided into inter-die variations and intra-die variations. Inter-die variations are variations that are constant within a die but vary from one die to another die on a wafer or in a wafer lot. Intra-die variations are variations that are present within a single die, meaning that a device/interconnect features vary between different locations on the same die. Intra-die variations result from equipment limitations or statistical effects in the fabrication process, such as statistical variations in the doping concentration.

Intra-die variation often exhibits spatial correlation. Devices that are close together have a higher probability of being alike than devices placed far apart. Intra-die variation can also have a deterministic component due to topologically dependent device processing, such as CMP (chemical-mechanical polishing) effects and optical proximity effects [95]. In some cases, such topological dependency can be accounted for directly in the analysis [135, 155]. However, the systematic variation cannot be analyzed until the layout is almost completed. Therefore, early in the design cycle, all intra-die variations are considered as random. In our analysis, we ignore the spatial correlations among intra-die variations. Both inter-die variation and intra-die variation can also change the load capacitances in a circuit, which in turn leads to the variation of the power consumption. We do not address the power variation due to the variation of the load capacitance because this source of variation can cause increase on some nodes and decrease on others, and on average does not lead to an increase of power dissipation of an optimized circuit.

4.2.2 Delay variation

Circuit delay is very sensitive to process variations because many factors can affect it. A simple first order derivation of the gate delay ([33], p. 89) is given by:

$$T_d = \frac{C_L \times V_{dd}}{I} = \frac{C_L \times V_{dd}}{\frac{\mu C_{ox}(W/L)}{2}(V_{dd} - V_t)^2} \tag{4.5}$$

where V_{dd} is the supply voltage, V_t is the threshold voltage, C_L is the load capacitance, $C_{ox} = \varepsilon_{ox}/t_{ox}$ is gate capacitance per unit area and T_d is the delay time. As we can see from the equation, even if each variable has only a very small change, the combined effect of all sources of variations can easily cause a dramatic change of the gate delay. Consider the delay variation to have a normal distribution with 10% standard deviation to mean ratio (σ/μ). The maximum-to-minimum delay ratio can be as large as $(\mu + 3\sigma)/(\mu - 3\sigma) = 1.86$. As technology shrinks to 90 nanometers and below, chips become much more difficult to manufacture. Intra-die process variations increase substantially at $90nm$ and even more for $65nm$. All these factors contribute to a relatively large delay variations and degradation of power saving by previous LP approaches [4, 170, 171].

Note that other than the process variation, the variation of environmental factors (such as power supply and temperature) can also cause the variation of delay. However, only physical factors, process variations, are considered in this dissertation because it is the dominating factor that affects the delays [145]. We assume for a small logic block, the temperature variation is not large enough to introduce additional intra-die delay variations. We also ignore the variation of supply voltage in our analysis.

4.2.3 Previous work

Since delays are very sensitive to process variations, most previous work on process variation uses static timing analysis (STA). Both deterministic STA and statistical STA have been proposed. In deterministic STA [82, 93], process variations have been modeled using the so-called case analysis. In this methodology, best-case, nominal and worst-case SPICE parameter sets are constructed and the timing analysis is performed several times, each time using one case file. Each execution of static timing analysis is therefore deterministic, meaning that the analysis uses deterministic delays for the gates and any statistical variation in the underlying silicon is hidden. The advantage of deterministic STA is its linear run time complexity with respect to the circuit size. However, with the continual scaling of feature sizes, the ability to control critical device parameters on a single die has become increasingly difficult. Using the worst-case parameters for intra-die variations therefore leads to very pessimistic analysis results since one assumes that all devices on a die have the worst-case characteristics.

Numerous statistical STA has been proposed [2, 13, 21, 27, 53, 56, 74, 94, 123, 124, 154] for above reasons. The disadvantage of statistical STA is that it has an underlying worst-case complexity that is exponential in the circuit size, which poses a fundamental obstacle to its practical application. This high run time complexity is the result of reconverging paths in the circuit, which cause correlations between path delays due to shared sections in paths. Therefore, most of the previous research on statistical STA concentrates on finding an accurate timing analysis and reducing the computation complexity. In this dissertation, we attempt both type of timing analyses in constructing our LP models.

In power optimization, not much work has been done on the effect of process (delay) variations. In [61, 188], the effect of process (delay) variation has been considered in voltage scaling or multiple V_{dd}/V_{th} optimization. Some other more related work is on gate sizing [80, 89, 128]. Jacob et al. [89] proposed a gate-sizing scheme under a statistical delay model. They consider the power optimization by minimizing the total gate size, which is equivalent to the total capacitance. Mani et al. [128] presented a statistical sizing approach that takes into account randomness in gate delays by formulating an efficient linear program. Similarly, they also tried to optimize the power by reducing the total gate size. Both of these approaches did not consider the reduction of glitch activity in the circuit. In another work, Hashimoto et al. [80] proposed a power optimization method by gate sizing, which considers glitch reductions in addition to the total capacitance and short circuit current. They use a statistical method to estimate the fluctuation of delay characteristics in the real circuit and an iterative heuristic algorithm to find the optimal gate sizes under delay constraint. However, this technique is based on the estimation of the glitch activity (power) at several sample points under skew fluctuation, where errors in power estimation undermines the optimization. Like all other heuristics algorithms, the global optimization of the solution is not guaranteed.

4.3 Delay model and implications

4.3.1 Random delay model

In our analysis, we adopt a random delay model. Delays are modeled as random variables instead of having deterministic values. We consider two basic types of process variation: inter-die variation and intra-die variation. Both of them have no dependence

on the device location. Variations are in terms of normalized values, i.e., σ/μ ratio. We propose the following model, where the gate delay $D_{total,i}$ of gate i is the algebraic sum of an inter-die gate delay $D_{inter,i}$ and an intra-die gate delay variation, $\Delta D_{intra,i}$:

$$D_{total,i} = D_{inter,i} + \Delta D_{intra,i} \tag{4.6}$$

where $D_{inter,i}$ and $\Delta D_{intra,i}$ are random variables with truncated normal distributions (truncated at 3σ). It's been showed in [27] that although gate delay is a nonlinear function of numerous variables (such as, gate oxide thickness t_{ox}, length and width of transistors, width of interconnect wire, etc.), a first order approximation of gate delay can be modeled by a Gaussian distribution. Same assumption is adopted in many other papers [21, 74, 123, 154]. Our adoption of truncated normal distributions reflects the fact that the gate delay in a chip cannot be more than a finite maximum value and less than a finite minimum value.

In this delay model, all $D_{inter,i}$ of gates on a die share one σ/μ ratio. For intra-die variations, each gate has a separate independent random variable $\Delta D_{intra,i}$. Both $D_{total,i}$ and $D_{inter,i}$ have the mean which is equal to the mean of the gate delay. The intra-die variation $\Delta D_{intra,i}$ has a mean of zero.

4.3.2 Effect of inter-die variation

In this section, we discuss the effect of inter-die variations to the switching power dissipation. The objective of the analysis is to prove that inter-die variations have negligible effect on the switching activity of a circuit. The inter-die gate delay $D_{inter,i}$ in Equation 4.6 can be further decomposed as the sum of its mean $D_{nom,i}$ and a zero mean random variable

$\Delta D_{inter,i}$. Therefore, we have the new gate delay model:

$$D_{total,i} = D_{nom,i} + \Delta D_{inter,i} + \Delta D_{intra,i} \tag{4.7}$$

The inter-die variation $\Delta D_{inter,i}$ has the same $\sigma/D_{nom,i}$ ratio for all gates on one die and its effect on the switching power dissipation depends on its effect on the switching activity in the circuit. Therefore, we first define the glitch-filtering probability and then show the effect of inter-die variations to the glitch-filtering probability in Theorem 1.

Definition. The glitch-filtering probability P_{glt} for a gate is the probability that signal arrival times t_1 and t_2 along paths 1 and 2 (assuming $t_1 \leq t_2$) have a difference smaller than the gate inertial delay d, i.e., $t_2 - t_1 < d$.

Theorem 1 *Assuming all inter-die variations $\Delta D_{inter,i}$ and all intra-die variations $\Delta D_{intra,i}$ have normal distributions, with zero mean ($\mu = 0$) and given $\sigma/D_{nom,i}$ ratios, while inter-die variations have the $\sigma/D_{nom,i}$ ratio equal to r, then the change of glitch-filtering probability ΔP_{glt} at a given gate due to inter-die variations is given by the equation:*

$$\Delta P_{glt} = \frac{1}{2}\left(\text{erf}\left(\frac{-k}{\sqrt{2}}\right) - \text{erf}\left(\frac{-k}{\sqrt{2+2(r\cdot k)^2}}\right)\right) \tag{4.8}$$

where $\text{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}dt$ *is the error function [189] and k depends upon the means and variances of the delays associated with the gate.*

Proof. First we consider the case that all $\Delta D_{inter,i} = 0$. For a given gate, signal arrival times t_1 and t_2 are sums of gate delays along paths. Therefore $t_1 = \sum_{i\in path_1} d_i$ and $t_2 = \sum_{j\in path_2} d_j$. The interval between t_1 and t_2 is then $t_2 - t_1 = \sum_{i\in path_1} d_i - \sum_{j\in path_2} d_j$, which

also has a normal distribution since it is a linear function of normal random variables. If we denote its mean as μ_{t_1,t_2} and standard deviation as σ_{t_1,t_2}, then the difference between gate delay d and signal arrival time interval $d-(t_2-t_1)$ is a random variable with mean $\mu_{d,t_1,t_2} = \mu_d - \mu_{t_1,t_2}$ and standard deviation $\sigma_{d,t_1,t_2} = \sqrt{\sigma_d^2 + \sigma_{t_1,t_2}^2}$.

For a normal random variable $X : N(\mu, \sigma^2)$, we have [189],

$$P[a < X \leq b] = \frac{1}{2}\text{erf}(\frac{\text{b}-\mu}{\sigma\sqrt{2}}) - \frac{1}{2}\text{erf}(\frac{\text{a}-\mu}{\sigma\sqrt{2}}) \tag{4.9}$$

Therefore the probability that $d-(t_2-t_1) > 0$ is then given by:

$$P_{glt} = \frac{1}{2} - \frac{1}{2}\text{erf}\left(\frac{-\mu_{d,t_1,t_2}}{\sigma_{d,t_1,t_2}\sqrt{2}}\right) = \frac{1}{2} - \frac{1}{2}\text{erf}\left(\frac{-k}{\sqrt{2}}\right) \tag{4.10}$$

where $k = \mu_{d,t_1,t_2}/\sigma_{d,t_1,t_2}$. $\sigma_{d,t_1,t2}$ is determined by the intra-die variations only.

When $\Delta D_{inter,i} \neq 0$ with $\sigma/D_{nom,i} = r$, each gate will have the same ratio of increase or decrease of delay and $d-(t_2-t_1)$ has a change $\mu_{d,t_1,t_2} \cdot \alpha$, where α is a normal random variable $N(0, r^2)$. In this case, $\sigma'_{d,t_1,t_2} = \sqrt{\sigma_{d,t_1,t_2}^2 + (r \cdot \mu_{d,t_1,t_2})^2}$ and the probability that $d-(t_2-t_1) > 0$ is then given by:

$$P'_{glt} = \frac{1}{2} - \frac{1}{2}\text{erf}\left(\frac{-\mu_{d,t_1,t_2}}{\sigma'_{d,t_1,t_2}\sqrt{2}}\right) = \frac{1}{2} - \frac{1}{2}\text{erf}\left(\frac{-k}{\sqrt{2+2(r\cdot k)^2}}\right) \tag{4.11}$$

from Equation 4.11 and 4.10 we obtain Equation 4.8.

■

Using Theorem 1, we can vary k and derive a range for ΔP_{glt} when r is given. Assuming $r = 0.15$, which represent a fairly large inter-die variation with a max/min ratio of 2.64, the value of ΔP_{glt} is plotted in Figure 4.3.

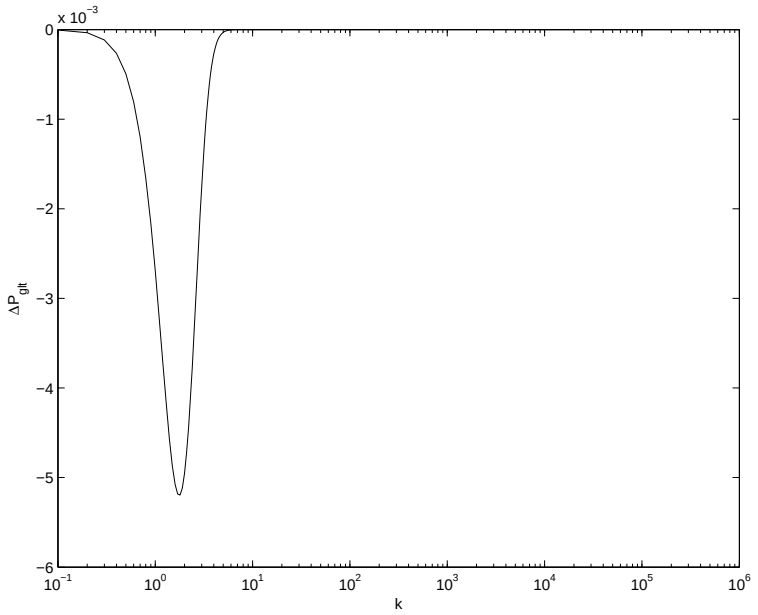


Figure 4.3: The range of ΔP_{glt} for $r = 0.15$ and varying k values.

We can see the absolute value of ΔP_{glt} is less than 0.6%, which means the change of glitch-filtering probability due to intra-die variations is negligible. Hence, we can conclude that inter-die variations have negligible effect on the switching activity of a circuit.

4.3.3 Process-variation-resistant design

Under our delay model, we can optimize a logic circuit and achieve a glitch-free design under process-variations by path balancing and glitch filtering. Such a design requires adjustment of delays and in some cases may increase the overall delay of the circuit. Unlike

that in [170], where a glitch-free design can always be obtained without increasing the overall circuit delay, the following theorem holds.

Theorem 2 *If the overall circuit delay is constrained not to increase, a glitch-free design under process variation cannot be guaranteed by path balancing and glitch filtering.*

Proof. This can be proved by contradiction. The example in Figure 4.4 shows a circuit with two longest paths with equal delays. Each gate has a minimum 1 unit delay and the critical delay for the circuit is 4 units. Without loss of generality, we can assume a 10% σ/μ ratio for delay variations of gates and therefore the path delay from A to C can be as small as $3 * (1 - 0.3) = 2.1$ units in an extreme case. Similarly, the path delay from B to C can be as large as $3 * (1 + 0.3) = 3.9$ units in another extreme case. Therefore the differential path delay at C will be $3.9 - 2.1 = 1.8$ units. Note that each gate has a 1 unit lower bound for its delay value. Under this circumstance, the only way to guarantee the suppression of glitches is to increase the gate delay of C. Since C is on the critical path, any increase in the delay of C will increase the overall circuit delay. ■

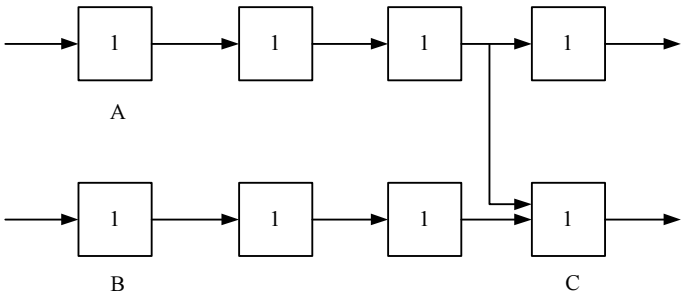


Figure 4.4: An example circuit for Theorem 2. Gates are represented with blocks with numbers indicating their inertial delays.

4.4 An LP Model based on worst-case timing analysis

The first LP model we propose is based on the worst-case analysis of timing. In this case, the earliest signal arrival time and latest signal arrival time are propagated under the worst-case condition of $\pm 3\sigma$ variations. Since inter-die variations have negligible effect on the switching activity in a circuit, we do not consider inter-die variations during the optimization step. The delay model is then:

$$D_{total,i} = D_{nom,i} + \Delta D_{intra,i} \tag{4.12}$$

where $D_{total,i}$ is a normal random variable with mean $\mu = D_{nom,i}$. In our analysis, we assume every gate has the same intra-die variation $r = \sigma/\mu$. However, if necessary, the extension of our LP model for varying intra-die parameter variation is straightforward.

Different from previous LP models [170, 171], we propose to use two timing windows for signal arrival times of a gate, the signal arrival times at the inputs of the gate and at the output of the gate. This is because the earliest and latest signals arriving at the gate will be delayed by the gate inertial delay. While in the worst-case analysis, the earliest arrival time and latest arrival time at the output of a gate are derived from the timing window at the inputs and two different worst-case values for the gate inertial delay. Therefore, the timing window at the output is always larger than the one at the input. To ensure a better optimization, it is necessary to apply the glitch-filtering constraint at the input. This rationale can be better explained with the help of Figure 4.5. We will describe our LP model in following sections.

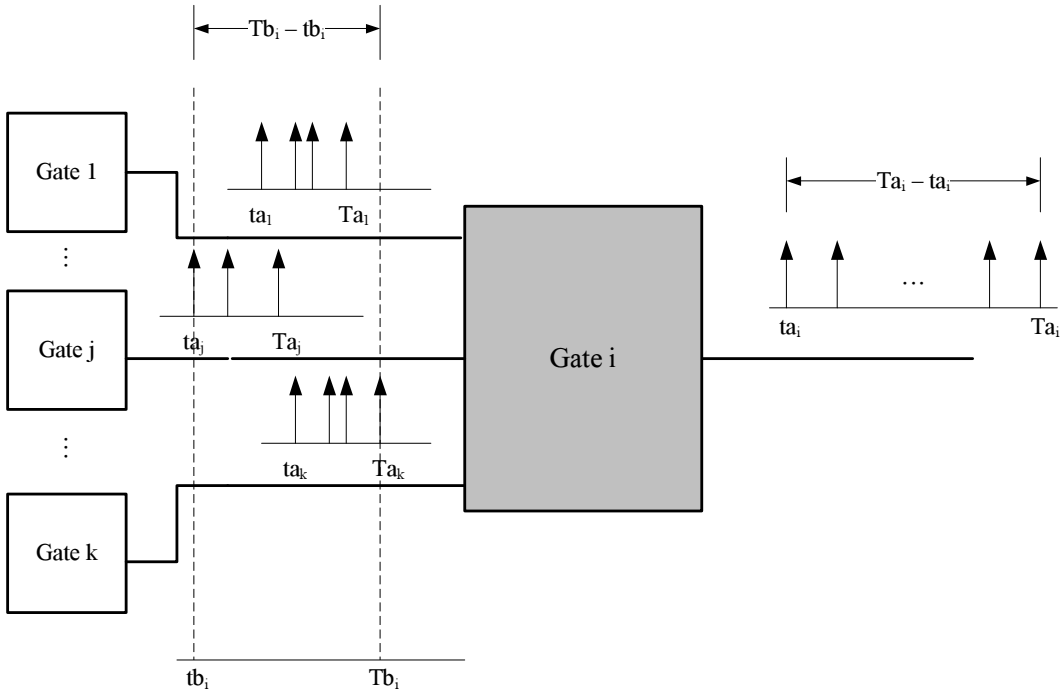


Figure 4.5: The illustration of the signal timing windows under the worst-case timing analysis.

4.4.1 Variables

As shown in Figure 4.5, variables are defined as follows,

- Ta_i: this is the *latest* time at which the *output* of gate i could have a signal transition event after the occurrence of an event at PIs.
- ta_i: this is the *earliest* time at which the *output* of gate i could have a signal transition event after the occurrence of an event at PIs.
- Tb_i: this is the *latest* time at which the *inputs* of gate i could have a signal transition event after the occurrence of an event at PIs.
- tb_i: this is the *earliest* time at which the *inputs* of gate i could have a signal transition event after the occurrence of an event at PIs.

- d_i: this is the *nominal* value of gate delay for gate i, $d_i = D_{nom,i}$. These values will be derived as the result of the LP model.

4.4.2 Constraints

Initial constraints

First, we need to set the initial conditions for all the variables. In this LP model, we have the constraints $d_i \geq 1$ for all gates and $d_i \geq 0$ for all buffers. $Ta_i \geq 0,\ ta_i \geq 0,\ Tb_i \geq 0,\ tb_i \geq 0$ for all gates and buffers. We also have the boundary condition for all PIs that $Ta_i = 0,\ ta_i = 0$. We assume that all signal events at PIs occur simultaneously.

Gate constraints

We need constraints to propagate timing windows through gates. Use Figure 4.5 as the example, we have gate constraints as follow:

$$\begin{aligned} Tb_i &\geq Ta_1; \\ Tb_i &\geq Ta_j; \\ Tb_i &\geq Ta_k; \\ tb_i &\leq ta_1; \\ tb_i &\leq ta_j; \\ tb_i &\leq ta_k; \\ Ta_i &= Tb_i + d_i \cdot (1 + 3r); \\ ta_i &= tb_i + d_i \cdot (1 - 3r); \end{aligned} \tag{4.13}$$

where r is the σ/μ ratio provided by the user indicating the intra-die variation. The first 3 constraints ensure that the value of Tb_i converges to the maximum value of Ta_1, Ta_j,

and Ta_k. The following 3 constraints ensure that the value of tb_i is the minimum value of ta_1, ta_j, and ta_k. As we can see from the last two constraints, the earliest and latest signal arrival times at the output of gate i are estimated using the worst-case variation of gate delay by the 3σ value.

Gate constraints are different for single-input-gates (inverters and buffers). Suppose we have a single-input-gate i with input from gate 1, the gate constraints are as follow:

$$\begin{aligned} Tb_i &= Ta_1; \\ tb_i &= ta_1; \\ Ta_i &= Tb_i + d_i \cdot (1 + 3r); \\ ta_i &= tb_i + d_i \cdot (1 - 3r); \end{aligned} \tag{4.14}$$

The timing window at the output is simply a propagation of timing window at inputs under the worst-case scenario.

Glitch-filtering constraints

Similar to the previous LP models, we also need to make sure each gate satisfies the glitch filtering condition in order to eliminate glitches. Therefore, for each gate i with more than one input, we have:

$$Tb_i - tb_i < d_i \cdot (1 - 3r); \tag{4.15}$$

where the worst-case value of gate delay is used to guarantee that the glitch filtering condition is always satisfied. For single-input-gates (inverters and buffers), such a glitch-filtering constraint is not applied to reduce unnecessary complexity.

Circuit delay constraints

Finally, we have to ensure that after the optimization, the optimized circuit is not slowed down beyond the specified upper bound on the maximum delay at the output. Therefore, for each gate i that produces a primary output, there is an additional constraint as:

$$Ta_i \leq D_{max}; \tag{4.16}$$

where D_{max} is a user specified maximum delay value.

4.4.3 Parameters

It is known that the worst-case timing analysis tends to be very pessimistic. Therefore, under the worst-case constraints, our LP model fails to give solutions in some cases when overall circuit delay is constrained. Therefore, besides the parameter r used to indicate the degree of intra-die variation and D_{max} the maximum delay requirement, we need an additional parameter to control the optimism in our LP model. We call it the optimism factor α. The glitch-filtering constraint (Inequality 4.15) is modified as:

$$Tb_i - tb_i < d_i \cdot (1 - 3r) \cdot \alpha \tag{4.17}$$

By giving a α value larger than 1.0, we assume that the actual glitch width in the circuit is smaller than the one derived from the worst-case timing analysis. When α is large, the glitch-filtering constraint is relaxed and our LP model is able to give solutions under a more stringent maximum delay constraint.

4.4.4 Objective function

Similar to the previous LP models [4, 170, 171], the objective is to minimize the number of buffers inserted in the circuit because the insertion of buffers increases the area of the circuit. Since this is a nonlinear objective, we adopt the same approach as in previous models to minimize the sum of delays of all buffers inserted.

4.5 Summary

In this chapter, we reviewed previous LP approach [170] by giving a background in linear programming and the detailed illustration of an LP model. The assumption that each gate has a fixed delay limits the application of the previous LP model under process variation. We analyzed the sources of process variations and constructed a random delay model, which contains inter-die and intra-die components. We showed that the effect of inter-die variations to switching activity (power) in a circuit is negligible. We prove that it is impossible to guarantee a glitch-free design under process variation if overall circuit delay is constrained. Our first process-variation-resistant LP model is then constructed based on a worst-case timing analysis. To control the optimism of the model, we adopt a factor α, which controls the relaxation of glitch-filtering constraints.

Chapter 5

LP Model Based on Statistical Timing Analysis

Since the worst-case timing analysis tends to be too pessimistic, an optimism factor was used to fine-tune the model in the previous chapter. In this chapter, we propose a different LP model based on statistical timing analysis. Some approximations have been made to the timing analysis to fit it in our LP model. In this model, the earliest and latest signal arrival times do not have deterministic values. They are random variables with distributions approximated as normal. This LP model also requires a fine-tuning by an optimism factor, but it is less pessimistic and therefore more generally applicable. We will first introduce the timing model based on a statistical static-timing-analysis (STA), and then we describe our LP model based on this new timing model.

5.1 Timing model

In statistical timing analysis, signal arrival times no longer have deterministic values but they are random variables. In our analysis, we assume that the earliest and latest signal arrival times, i.e., the earliest and latest time at which a signal transition event can occur after an occurrence of event at PIs, are random variables with truncated normal distributions. As we will see later in this section, this is an approximation for the real distribution even though gate delays are assumed to have normal distributions. This approximation could lead to degradations of timing accuracy in some cases. However, this assumption facilitates the propagation of the signal arrival times and timing windows throughout the circuit, making statistical STA possible in a linear programming approach.

5.1.1 Time variables

We model the earliest and latest signal arrival times t and T as random variables with (truncated) normal distributions. Random variable t has a mean μ_t and standard deviation σ_t. Similarly, T has a mean μ_T and standard variation σ_T. Therefore, the timing model illustration in Figure 4.5 has to be modified to indicate that all signal arrival times t and T are random variables.

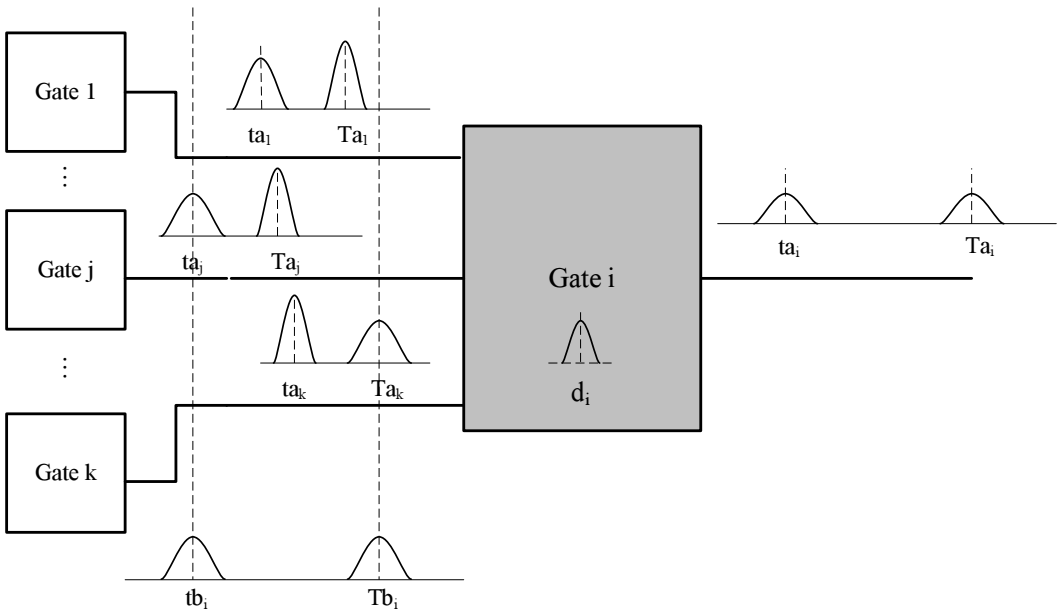


Figure 5.1: An illustration of the signal timing windows in statistical timing analysis

As discussed above, gate delays are assumed to be random variables with (truncated) normal distributions. Therefore, the signal arrival times at the output of a gate i, e.g., ta_i, Ta_i, can be obtained by a simple summation of signal arrival times at the inputs of gate i and gate inertial delay d_i:

$$\begin{aligned} ta_i &= tb_i + d_i \\ Ta_i &= Tb_i + d_i \end{aligned} \tag{5.1}$$

In our statistical timing analysis, we inherit notations from the previous timing model of Chapter 4. However, we should clarify the differences in their meaning. In the previous timing model, t and T indicate the worst-case value of signal arrival times and therefore are deterministic values. However, in the statistical timing model, t and T represent the earliest and latest signal arrival times, which are random variables. Similarly, notation d_i in the previous model was the nominal value of the inertial delay for gate i; but in the statistical timing model, it represents a random gate delay with a (truncated) normal distribution.

5.1.2 Maximum and minimum statistics

To derive the signal timing window at the inputs of a gate, maximum and minimum statistics are needed. Using Figure 5.1 as the example, the earliest and latest signal arrival times at the inputs are then:

$$\begin{aligned} tb_i &= Min(ta_1, ta_j, ta_k) \\ Tb_i &= Max(Ta_1, Ta_j, Ta_k) \end{aligned} \tag{5.2}$$

where $Min()$ and $Max()$ represent the minimum and the maximum of a set of random variables. In our analysis, all t and T are assumed to be spatially independent. In a real circuit, a signal propagated through reconvergent paths can lead to spatial correlations. However, it is impossible to deal with such correlations in our linear programming approach since that could mean an exponential increase in complexity. Even in the existing statistical STA work, spatial independence is often assumed in order to reduce the computation complexity. It has been proved in [2] that such an approach that neglects spatial correlations will lead to an upper bound of the resulting distribution.

Previous work

In our analysis, we assume that all the signal arrival times t and T are random variables with (truncated) normal distributions. However, the resulting variable from above maximum or minimum operations is not necessarily distributed as a normal random variable. In [21], Berkelaar has shown that the resulting distribution from the maximum operation of two random variables, each having a normal distribution, is very similar to but not necessarily same as a normal distribution. Based on the statistical timing analysis in [21], Jacob and Berkelaar have proposed a method that approximates the result of the maximum or minimum operation with a random variable in normal distribution [89]. The output mean and standard deviation can be expressed as a closed-form formula of input means and standard deviations.

The approximation proposed in [89] is as follows. For maximum operation $C = Max(A, B)$, where $A = N(\mu_A, \sigma_A^2)$ and $B = N(\mu_B, \sigma_B^2)$ are random variables with normal distributions, the output C can be approximated as $N(\mu_C, \sigma_C^2)$, where μ_C and σ_C^2 are functions of μ_A, μ_B, σ_A, and σ_B given by the following equations:

$$\mu_C = \frac{\sqrt{\sigma_A^2+\sigma_B^2}}{\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\mu_A-\mu_B}{\sqrt{\sigma_A^2+\sigma_B^2}}\right)^2} + \mu_A \phi\left(\frac{\mu_A-\mu_B}{\sqrt{\sigma_A^2+\sigma_B^2}}\right) + \mu_B \phi\left(\frac{\mu_B-\mu_A}{\sqrt{\sigma_A^2+\sigma_B^2}}\right) \tag{5.3}$$

in which $\phi(x)$ is given by

$$\phi(x) = \int_{-\infty}^{x} e^{-\frac{1}{2}u^2} du \tag{5.4}$$

Variance σ_C^2 is given by,

$$\begin{aligned}\sigma_C^2 = \quad & (\mu_A + \mu_B)\frac{\sqrt{\sigma_A^2+\sigma_B^2}}{\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{\mu_A-\mu_B}{\sqrt{\sigma_A^2+\sigma_B^2}}\right)^2} + \\ & (\sigma_A^2 + \mu_A^2)\,\phi\left(\frac{\mu_A-\mu_B}{\sqrt{\sigma_A^2+\sigma_B^2}}\right) + \\ & (\sigma_B^2 + \mu_B^2)\,\phi\left(\frac{\mu_B-\mu_A}{\sqrt{\sigma_A^2+\sigma_B^2}}\right) - \mu_C^2\end{aligned} \tag{5.5}$$

Our approximation

It would been nice if we could use the above approximation in our LP model, which has already been proved effective [89]. However, as we can see from above equations, this is a very complex operation that requires multiplication, division, square root, integrations, etc. These non-linear operations are prohibited in a linear programming approach because they will convert the problem to a non-linear program. The optimization by a non-linear program is highly computation intensive as compared to that by a linear program. In addition, the global optimization of the solution may not be guaranteed. Therefore, to accommodate the statistical timing analysis into a LP model is a challenging task, and we have to find simpler expressions to approximate the resulting distribution with a reasonable accuracy.

In our analysis, given random variables $A = N(\mu_A, \sigma_A^2)$ and $B = N(\mu_B, \sigma_B^2)$ with normal distributions, we approximate $C = Max(A, B)$ as a random variable with normal distribution $N(\mu_C, \sigma_C^2)$, where μ_C, σ_C are functions of μ_A, μ_B, σ_A, and σ_B. We have,

$$\begin{aligned}&\mu_C = Max(\mu_A, \mu_B);\\ &\sigma_C = \tfrac{1}{3}\cdot(Max(\mu_A + 3\sigma_A, \mu_B + 3\sigma_B) - \mu_C)\end{aligned} \tag{5.6}$$

Similarly, we approximate $D = Min(A, B)$ as a random variable with normal distribution $N(\mu_D, \sigma_D^2)$, where μ_D, σ_D are functions of μ_A, μ_B, σ_A, and σ_B,

$$\begin{aligned} \mu_D &= Min(\mu_A, \mu_B); \\ \sigma_D &= \tfrac{1}{3} \cdot (\mu_D - Min(\mu_A - 3\sigma_A, \mu_B - 3\sigma_B)) \end{aligned} \tag{5.7}$$

We believe that the above approximations suit our needs. First, the approximations are in very simple forms that do not require any non-linear operation. Therefore, we can incorporate the statistical timing analysis into our linear program. It results in a LP model that can be solved efficiently and the solution is guaranteed to be globally optimal. Second, the error in timing does not directly translate into errors in the power optimization. Our objective is not to derive a precise timing analysis but to use the timing analysis for glitch reduction. Small differences in timing analysis do not contribute significantly to the optimality of the final solution. As we will see later, the critical delay of the circuit will be not be affected by this timing analysis because it will be guaranteed by constraints using the worst-case condition.

Approximation errors

Although the above approximation suits our needs for power optimization, it is necessary to analyze the error it introduces in the timing analysis. Let us consider the example in Figure 5.2, where the cumulative distribution function (CDF) for two normally distributed random variables A and B are plotted. The definitions for CDFs are,

$$\begin{aligned} CDF_A(x) &= P[A \leq x] \\ CDF_B(x) &= P[B \leq x] \end{aligned} \tag{5.8}$$

where x is the variable of the CDF. Assuming that A and B are mutually independent, the resulting CDF for $C = Max(A, B)$ is simply the product of the CDF, for A and B,

$$\begin{aligned} CDF_C(x) &= P[C \leq x] = P[Max(A, B) \leq x] = P[A \leq x] \cdot P[B \leq x] \\ &= CDF_A(x) \cdot CDF_B(x) \end{aligned} \tag{5.9}$$

In case that $(\mu_B - \mu_A) > 3 \cdot (\sigma_A + \sigma_B)$,

$$CDF_C(x) \approx CDF_B(x) \tag{5.10}$$

Similarly, when $(\mu_A - \mu_B) > 3 \cdot (\sigma_A + \sigma_B)$,

$$CDF_C(x) \approx CDF_A(x) \tag{5.11}$$

The resulting CDF of C given by our approximation has

$$\mu_C = \mu_B; \quad \sigma_C = \sigma_B; \quad \text{when } (\mu_\text{B} - \mu_\text{A}) > 3 \cdot (\sigma_\text{A} + \sigma_\text{B}); \tag{5.12}$$

or

$$\mu_C = \mu_A; \quad \sigma_C = \sigma_A; \quad \text{when } (\mu_\text{A} - \mu_\text{B}) > 3 \cdot (\sigma_\text{A} + \sigma_\text{B}); \tag{5.13}$$

Therefore, our approximation does not lead to any significant error in these two cases.

However, when A and B have $|\mu_A - \mu_B| \leq 3 \cdot (\sigma_A + \sigma_B)$, our approximation begins to deviate more from the actual CDF_C. In Figure 5.2, the actual CDF_C is plotted by a dashed line. Our approximation of CDF_C is shown by a thick solid line as it overlaps with

CDF_B. This approximation error becomes maximum when $\mu_A = \mu_B$. In Chapter 7, we will illustrate the accuracy of our statistical timing analysis by Monte-Carlo simulation.

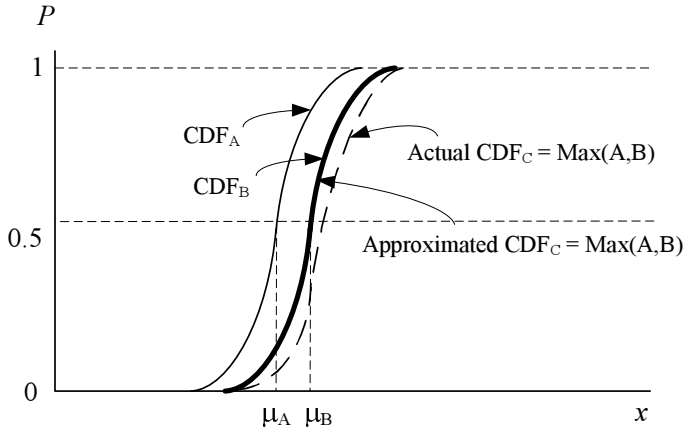


Figure 5.2: The illustration of the maximum operation of two random variables A and B. Cumulative distribution functions are plotted, where actual CDF for $Max(A, B)$ is plotted in the dashed line.

5.2 An LP model based on statistical timing analysis

We construct our second linear programming model based on the statistical STA described above. Few other approximations are made to incorporate the statistical STA into our model.

5.2.1 Variables

Since signal arrival times and gate delays are random variables with (truncated) normal distributions, they can be described by mean μ and standard deviation σ. Therefore, each time variable and gate delay variable is characterized by its μ and σ. Similar to our first LP model in the previous chapter, we assume that every gate has the same intra-die variation

$r = \sigma/\mu$. Inter-die variations are not considered in this optimization. Variables are defined as follows,

- Variables for Ta_i: this is the *maximum statistic* of the *latest* times at the *output* of gate i to have a signal transition event after the occurrence of an event at the PIs. Ta_i is assumed to be a random variable with a truncated normal distribution.
  - μ_{Ta_i}: this is the *mean* of Ta_i.
  - σ_{Ta_i}: this is the *std. dev.* of Ta_i.
- Variables for ta_i: this is the *minimum statistic* of the *earliest* times at the *output* of gate i to have a signal transition event after the occurrence of an event at the PIs. ta_i is assumed to be a random variable with a truncated normal distribution.
  - μ_{ta_i}: this is the *mean* of ta_i.
  - σ_{ta_i}: this is the *std. dev.* of ta_i.
- Variables for Tb_i: this is the *maximum statistic* of the *latest* times at the *inputs* of gate i to have a signal transition event after the occurrence of an event at the PIs. Tb_i is assumed to be a random variable with a truncated normal distribution.
  - μ_{Tb_i}: this is the *mean* of Tb_i.
  - σ_{Tb_i}: this is the *std. dev.* of Tb_i.
  - T_{Tb_i}: this is a auxiliary variable used for the $Max()$ operation of signal arrival times.

- Variables for tb_i: this is the *minimum statistic* of the *earliest* times at the *inputs* of gate i to have a signal transition event after the occurrence of an event at the PIs. tb_i is assumed to be a random variable with a truncated normal distribution.
  - μ_{tb_i}: this is the *mean* of tb_i.
  - σ_{tb_i}: this is the *std. dev.* of tb_i.
  - t_{tb_i}: this is a auxiliary variable used for the $Min()$ operation of signal arrival times.
- Variables for the timing window, $W_i = Tb_i - tb_i$: this is an auxiliary variable of the timing window *statistic*, indicating the difference between Tb_i and tb_i.
  - μ_{W_i}: the *mean* of W_i.
  - σ_{W_i}: the *std. dev.* of W_i
- Variables for d_i: this is the gate delay *statistic* for gate i. Random variable d_i is assumed to have a truncated normal distribution.
  - μ_{d_i}: this is the *nominal value* of the gate delay for gate i. The value will be derived as the result of the LP model.

5.2.2 Constraints

Similar to our first LP model, we have initial constraints, gate constraints, glitch-filtering constraints and maximum delay constraints.

Initial constraints

The initial constraints for all the variables are similar to that in our first LP model. We have the constraints $\mu_{d_i} \geq 1$ for all gates and $\mu_{d_i} \geq 0$ for all buffers. For all gates and buffers,

$$\begin{aligned} &\mu_{Ta_i} \geq 0; \quad \sigma_{Ta_i} \geq 0; \\ &\mu_{ta_i} \geq 0; \quad \sigma_{ta_i} \geq 0; \\ &\mu_{Tb_i} \geq 0; \quad \sigma_{Tb_i} \geq 0; \quad T_{Tb_i} \geq 0; \\ &\mu_{tb_i} \geq 0; \quad \sigma_{tb_i} \geq 0; \quad t_{tb_i} \geq 0; \\ &\mu_{W_i} \geq 0; \quad \sigma_{W_i} \geq 0; \end{aligned} \tag{5.14}$$

We also have the boundary conditions for all PIs, assuming no variation in signal arrival times at PIs.

$$\begin{aligned} &\mu_{Ta_i} = 0; \quad \sigma_{Ta_i} = 0; \\ &\mu_{ta_i} = 0; \quad \sigma_{ta_i} = 0; \end{aligned} \tag{5.15}$$

Gate constraints for multi-input-gates

The propagation of timing window is different from our first LP model. Since all the signal arrival times are random variables, we propagate them in terms of the mean μ and standard deviation σ.

Gate constraints at the inputs: Consider a two-input gate i. Assuming that gate i has two inputs from gate 1 and 2, the propagation of time variables from the inputs of gate

i follows the constraints,

$$\begin{aligned}
&\mu_{Tb_i} \geq \mu_{Ta_1};\\
&\mu_{Tb_i} \geq \mu_{Ta_2};\\
&T_{Tb_i} \geq \mu_{Ta_1} + 3\sigma_{Ta_1};\\
&T_{Tb_i} \geq \mu_{Ta_2} + 3\sigma_{Ta_2};\\
&\sigma_{Tb_i} = (T_{Tb_i} - \mu_{Tb_i})/3;
\end{aligned} \tag{5.16}$$

and

$$\begin{aligned}
&\mu_{tb_i} \leq \mu_{ta_1};\\
&\mu_{tb_i} \leq \mu_{ta_2};\\
&t_{tb_i} \leq \mu_{ta_1} - 3\sigma_{ta_1};\\
&t_{tb_i} \leq \mu_{ta_2} - 3\sigma_{ta_2};\\
&\sigma_{tb_i} = (\mu_{tb_i} - t_{tb_i})/3;
\end{aligned} \tag{5.17}$$

Constraint set 5.16 ensures that μ_{Tb_i} settles at $Max(\mu_{Ta_1}, \mu_{Ta_2})$ and T_{Tb_i} settles at $Max(\mu_{Ta_1} + 3\sigma_{Ta_1}, \mu_{Ta_2} + 3\sigma_{Ta_2})$. Therefore, this set of constraints realizes the approximation of $Tb_i = Max(Ta_1, Ta_2)$ from Equation 5.6. Similarly, constraint set 5.17 realizes the approximation of $tb_i = Min(ta_1, ta_2)$ from Equation 5.7.

Gate constraints at the output: At the output of gate i, we determine the time variables by adding the signal arrival times at the inputs with the gate delay d_i, i.e., $Ta_i = Tb_i + d_i$ and $ta_i = tb_i + d_i$. We have the following relations:

$$\begin{aligned}
&\mu_{Ta_i} = \mu_{Tb_i} + \mu_{d_i}; \quad \sigma_{Ta_i} = k(\sigma_{Tb_i} + r \cdot \mu_{d_i});\\
&\mu_{ta_i} = \mu_{tb_i} + \mu_{d_i}; \quad \sigma_{ta_i} = k(\sigma_{tb_i} + r \cdot \mu_{d_i});
\end{aligned} \tag{5.18}$$

where r is the σ/μ ratio for all d_i and $r \cdot \mu_{d_i}$ gives the σ_{d_i} value.

In essence, we have applied a linear approximation in the derivation of σ. Precisely, $\sigma_{Ta_i} = \sqrt{\sigma_{Tb_i}^2 + (r \cdot \mu_{d_i})^2}$. However, we cannot do such nonlinear operations in a linear program. To solve this problem, we adopt a linear approximation of the type $\sigma_{Ta_i} = k(\sigma_{Tb_i} + r \cdot \mu_{d_i})$.

Given two variables A and B $(A, B \geq 0)$, the range for $\sqrt{A^2 + B^2}$ can be determined as follows. First, we have

$$\sqrt{A^2 + B^2} \leq \sqrt{A^2 + B^2 + 2AB} = A + B \tag{5.19}$$

Then, since $A^2 + B^2 - 2AB \geq 0$, we have $A^2 + B^2 \geq 2AB$. Therefore, $2(A^2 + B^2) \geq A^2 + B^2 + 2AB$. So, we have

$$\sqrt{A^2 + B^2} \geq \frac{\sqrt{A^2 + B^2 + 2AB}}{\sqrt{2}} = \frac{A + B}{\sqrt{2}} \tag{5.20}$$

Considering $\frac{A+B}{\sqrt{2}} \leq \sqrt{A^2 + B^2} \leq A + B$, the range for k is $k \in (\frac{\sqrt{2}}{2}, 1)$. Although such approximation will lead to further errors in the timing analysis, we found our LP model works effectively under this approximation. In our implementation, we adopt $k = 0.85$, the mid-point of the range for k, to minimize worst-case approximation errors.

Gate constraints for single-input-gates

For single-input-gates, such as inverters and buffers, the gate constraints are much simpler. For gate i with only one input from gate 1, the gate constraints are:

$$\begin{aligned} &\mu_{Ta_i} = \mu_{Ta_1} + \mu_{d_i}; \quad \sigma_{Ta_i} = k(\sigma_{Ta_1} + r \cdot \mu_{d_i}); \\ &\mu_{ta_i} = \mu_{ta_1} + \mu_{d_i}; \quad \sigma_{ta_i} = k(\sigma_{ta_1} + r \cdot \mu_{d_i}); \end{aligned} \tag{5.21}$$

Note that there is no constraint at the input of a single-input-gate. The timing window is propagated directly to the output of the gate.

Glitch-filtering constraints

To ensure the minimum dynamic power, glitch filtering condition must be satisfied at each gate with more inputs than one. This constraint can be expressed as $d_i > Tb_i - tb_i$ or $d_i - (Tb_i - tb_i) > 0$. Since d_i, Tb_i and tb_i are random variables with normal distributions, $d_i - (Tb_i - tb_i)$ also has a normal distribution. Therefore, we have the glitch-filtering constraints for each gate i with more inputs than one:

$$\begin{aligned} &\mu_{W_i} = \mu_{Tb_i} - \mu_{tb_i}; \\ &\sigma_{W_i} = k(\sigma_{Tb_i} + \sigma_{tb_i}); \\ &\mu_{d_i} - \mu_{W_i} > 3 \cdot k(\sigma_{W_i} + r \cdot \mu_{d_i}); \end{aligned} \tag{5.22}$$

The first two equalities derive the mean and standard deviation for $W_i = Tb_i - tb_i$. The last inequality ensures that $\mu_{d_i - W_i} - 3\sigma_{d_i - W_i} > 0$ and guarantees $d_i - W_i > 0$.

Maximum delay constraint

Finally, the circuit has to satisfy the maximum delay constraint to ensure that the optimized circuit is not slowed down beyond the specified limit. For each gate i producing a primary output, there is a constraint:

$$\mu_{Ta_i} \cdot (1 + 3r) \leq D_{max}; \tag{5.23}$$

where D_{max} is a user specified maximum delay value. Here we ensure the circuit critical delay is within the range for worst-case variations.

5.2.3 Parameters

Although the LP model proposed in this chapter is less pessimistic, it is desirable that we control the optimism in the model. Under tight (inflexible) delay constraints, it may be sometimes impossible to find a solution. Therefore, except the parameter r used to indicate the degree of intra-die variation and D_{max} for the maximum delay requirement, we adopt an optimism factor α to control the optimism of the model. Therefore, we modify the glitch-filtering constraint in constraint set 5.22 to:

$$\mu_{d_i} - \mu_{W_i} > 3 \cdot k(\sigma_{W_i} + r \cdot \mu_{d_i}) * \alpha; \tag{5.24}$$

By choosing an α less than 1, we assume that the actual glitch width is smaller than the one derived from the statistical timing analysis. When α is small, the glitch-filtering constraint is relaxed and our LP model is able to give solutions under a more stringent maximum delay constraint. When $\alpha = 0$, our LP model reduces to the LP model in [170].

5.2.4 Objective function

Similar to previous LP models, the objective is still to minimize the number of buffers inserted in the circuit. Since this is a nonlinear objective, we adopted the same approach as that in previous models to minimize the sum of delays of all buffers inserted.

5.3 Summary

In this chapter, a new LP model is constructed based on a statistical timing analysis. In our statistical timing analysis, all the signal arrival times and gate delays are treated as random variables in truncated normal distribution. To propagate the earliest and the latest signal arrival times, we propose a linear approximation method for the minimum and maximum statistics. Based on our statistical timing analysis, we define variables for mean and standard deviations of signal arrival times and gate delays. Signal arrival times and timing windows are propagated in terms of mean and standard deviations. Glitch-filtering constraints are also implemented in a statistical manner to guarantee that the glitch filtering conditions are met under the delay variations. An optimism factor α is used to control the optimism of the model in order to ensure solutions under tight maximum delay requirements, where a totally glitch-free design may be impossible.

Chapter 6

Input-Specific Optimization

In this chapter, we propose a new way of circuit optimization, the input-specific optimization. In an input-specific optimization, a circuit is optimized only with respect to a specific set of vectors (patterns), typically, the functional vectors of the circuit. This customized optimization can result in a circuit with almost the same reduction of power dissipation but with lower overhead in terms of number of buffers inserted. In this chapter, we first explain our motivation then give the concepts of *glitch-generation pattern* and *glitch-generation probability*. Utilizing the measure of glitch-generation probability, we can customize the optimization of a circuit according to the given set of input vectors. This input-specific optimization technique is first applied to the previous LP model [170] under no process variation. It is then augmented into our process-variation-resistant LP model proposed in Chapter 5.

6.1 Motivation

Previous LP modeling [4, 170, 171] considers the optimization of the circuit in the *worst-case*. A timing window of signal arrival time $[t_i, T_i]$ is propagated throughout the circuit, where t_i is the earliest arrival time and T_i is the latest arrival time for gate i. There is a constraint $d_i > T_i - t_i$ for each gate inertial delay d_i. Therefore, the LP solution ensures that the circuit is free from glitch for *any* input vector sequence. However, we observe, this worst-case optimization may have introduced too much redundancy to the solution. For a circuit where the total propagation delay is restricted, the LP solution may require insertion

of a large number of buffers. As we know, the insertion of buffers is costly and should be kept as less as possible because it either increases the total power dissipation of the circuit (assuming conventional buffers) or the total area of the circuit (assuming resistance type of buffers).

In reality, the above worst-case optimization may mean some overdesign. We may not need the circuit to be optimized for *all* possible input sequences. On the contrary, we may only want the circuit be optimized for the set of input sequences that will be applied to the circuit, for example, the functional vectors while it is working. These input sequences can be a highly biased set depending on the circuit environment. Optimization of a circuit specific to such vector sequences ensures that the optimized circuit maintains the low power dissipation under the given system environment. At the same time, we are able to achieve a better solution with reduced overhead because the optimization is more customized.

6.2 Glitch generation

In our input-specific optimization, we suggest to relax the constraints for gates where glitches are unlikely or impossible. First, it is necessary for us to understand how a glitch is generated. In this section, we discuss the generation of glitches and introduce the concepts of *glitch-generation pattern* and *glitch-generation probability*.

6.2.1 Glitch-generation pattern

As mentioned in Section 2.3.1, *glitches/hazards* refer to the spurious transitions at the output due to the differential path delays. Two factors are essential to glitch generations, i.e., transitions and path delays. Here, we define a *glitch-generation pattern* for a gate as

the input vector pair that can potentially generate a glitch at the output of a gate if the input path delays are varied.

As shown in Figure 6.1, glitch-generation patterns for a two-input AND/OR gate are those vector pairs that produce two opposite transitions on the different inputs. However, for a two-input XOR gate, a glitch can be generated potentially as long as both inputs have transitions.

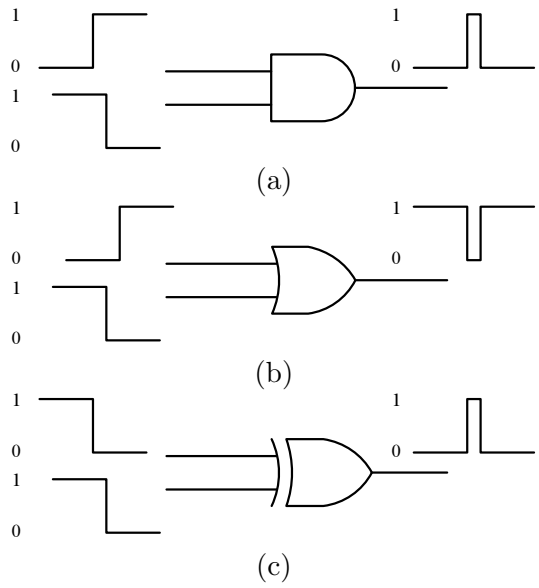


Figure 6.1: The glitch-generation patterns for two-input gates: (a) a glitch-generation pattern for a two-input AND gate, (b) a glitch-generation pattern for a two-input OR gate, (c) a glitch-generation pattern for a two-input XOR gate.

For a gate with more than two inputs, a glitch can be generated potentially only if there is no controlling value at any input of the gate. Therefore, the glitch-generation patterns for a multi-input AND gate will be those vector pairs that produce opposite transitions at any two inputs and no constant zeros at any other input. The glitch-generation patterns for a multi-input OR gate will be those vector pairs that produce opposite transitions at any two inputs and no constant ones at any other input. Since there is no controlling value for

an XOR gate, the glitch-generation patterns for a multi-input XOR gate are those vector pairs that produce either rising or falling transition on at least two inputs. Figure 6.2 shows the effect of a controlling value on glitch generation from different gates.

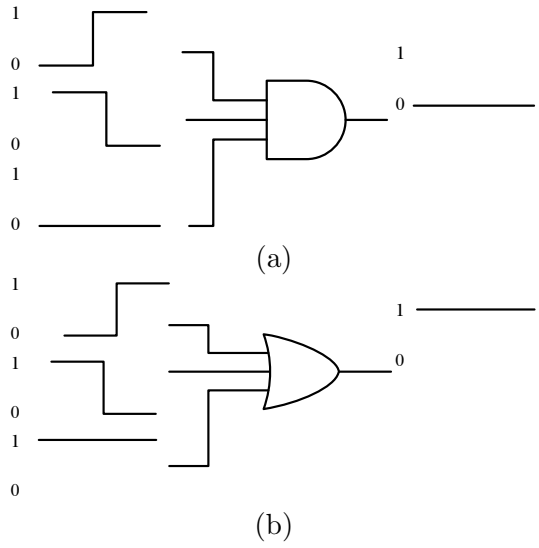


Figure 6.2: The effect of the controlling value to glitch generation for multi-input gates: (a) the controlling value for an AND gate; (b) the controlling value for an OR gate. A glitch cannot be generated if any input of the gate has a constant controlling value.

6.2.2 Glitch-generation probability

We define *glitch-generation probability* P_g for a gate as the probability that a glitch-generation pattern occurs at the input of the gate. By the word "occur", we mean that the *steady-state* signal values for two consecutive clock periods at inputs of the gate match a glitch-generation pattern for this gate.

Under a specific set of N input vectors, glitch-generation probability for each gate can be obtained through zero delay logic simulation of the circuit. Denote the number of times a glitch-generation pattern occurs at the input of gate i by $N_g[i]$, the glitch-generation

probability for gate i, $P_g[i]$, can be calculated as

$$P_g[i] = \frac{N_g[i]}{N} \tag{6.1}$$

6.3 Input-specific optimization

With the measure of glitch-generation probability, we can selectively relax the constraints for gates where glitches are unlikely to occur. In this section, we describe how we utilize these measures to perform an input-specific optimization.

6.3.1 Application to the previous LP model

First, we apply the input-specific optimization to the previous LP model [170] under no process variation. This helps to illustrate the basic technique and the concept. An input-specific optimization will still be able to achieve a glitch-free circuit under the given set of input sequence. However, it can significantly reduce the redundancy by relaxing unnecessary gate constraints.

Static optimization

Our input-specific optimization is a "static" analysis. It means that only probabilities of glitch generations are characterized as the basis for relaxations of constraints. As shown in Figure 2.12, glitches in a practical circuit can be either generated at a gate (static hazards) or propagated from the previous stage of the circuit (dynamic hazards). Our definition of glitch-generation probability only captures potential static glitches and ignores possible glitches propagated from the previous stage. Due to the underlying difficulty

and complexity related to the analysis for propagated glitches, we only adopt the glitch-generation probability to approximate the chance that a glitch can be produced if no proper optimization is done.

This ignorance of glitch propagation could mean problems in some cases. As the example shown in Figure 6.3, the steady-state values for the two-input AND gate are 01 and 00. It does not match any glitch-generation pattern of a two-input AND gate. However, a glitch at the input is able to propagate to the output of the gate and produce a propagated glitch.

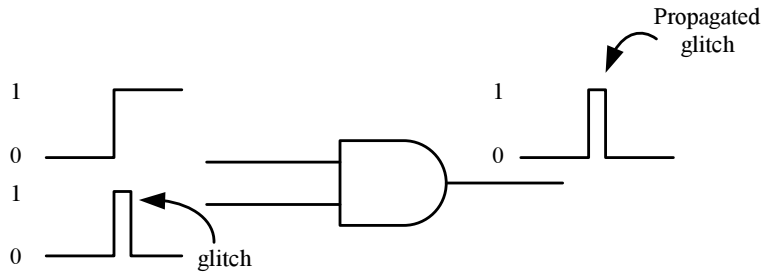


Figure 6.3: The limitation of our glitch-generation probability. Propagated glitches are not captured by our definition of glitch-generation probability.

From the above example, we understand the accuracy of the glitch-generation probability to represent the chance that a glitch can be produced is strongly affected by the ratio of propagated glitches. Only when propagated glitch does not exist or has a negligible probability, can our glitch-generation probability represent the chance correctly. In our relaxation of constraints, we adopt the assumption that no (or a negligible amount of) glitches can be propagated from the previous stages of the circuit. Our input-specific optimization method is designed specifically to ensure this assumption is valid in most cases.

Selective relaxation

Under the assumption that no glitch can be propagated throughout the circuit, glitch-generation probability of a gate reflects the chance that a glitch can be produced at output of the gate if no proper path balancing or glitch filtering is done. For gates with glitch-generation probability equal to zero, a glitch-generation pattern will never be produced at the gate inputs by the given set of primary input sequence. It also means that a glitch will never occur no matter how path delays change. Under this circumstance, we consider the relaxation of gate constraints by removing the glitch-filtering constraint.

As we know, the original glitch-filtering constraint for gate i [170] has the form

$$d_i > T_i - t_i \tag{6.2}$$

In the input-specific optimization, it is modified to

$$d_i > (T_i - t_i) \cdot \beta_i \tag{6.3}$$

where $\beta_i \in \{0, 1\}$ is a constant and is determined by the glitch-generation probability of the gate i:

$$\beta_i = \begin{cases} 0 & \text{if } P_g[i] = 0 \\ 1 & \text{if } P_g[i] > 0 \end{cases} \tag{6.4}$$

Essentially, the glitch-filtering constraints for gates are removed selectively according to the glitch-generation probability. Note that this selective relaxation method does not change the glitch-free property (i.e., no glitches are generated) of the resulting circuit since any potential glitches are always filtered out at all other gates where glitch-generation

probability is not zero. However, this input-specific optimization can result in a circuit with lower overhead because the relaxation of constraints. Our assumption that no glitch is propagated throughout the circuit is always true since glitches are suppressed (i.e., never generated) wherever they were possible.

Generalized relaxation

The above selective relaxation can be further generalized to allow even greater relaxation of constraints. The generalized relaxation, however, does not guarantee that the circuit will be totally glitch-free. However, it provides designers a trade off between power dissipation and number of buffers inserted under a given critical delay requirement. In this generalized relaxation, we consider replacing the step function in Equation 6.4 with

$$\beta_i = 1 - e^{-P_g[i]/\tau} \tag{6.5}$$

Here, β_i is an exponential function of the glitch-generation probability $P_g[i]$ with a tuning factor τ. The function β_i with different τ is illustrated in Figure 6.4.

In this generalized relaxation, glitch-filtering constraints are relaxed according to the glitch-generation probability. The adoption of an exponential function has two advantages. First, for gates where glitches are more likely to occur, the glitch-filtering constraint is enforced ($\beta_i = 1$). Second, for gates where glitches are less likely to occur, the glitch-filtering constraint is relaxed accordingly. The fast rising slope of the exponential function for small $P_g[i]$ ensures that only a small number of glitches will be propagated to the next stage, which supports our assumption about propagation of glitches.

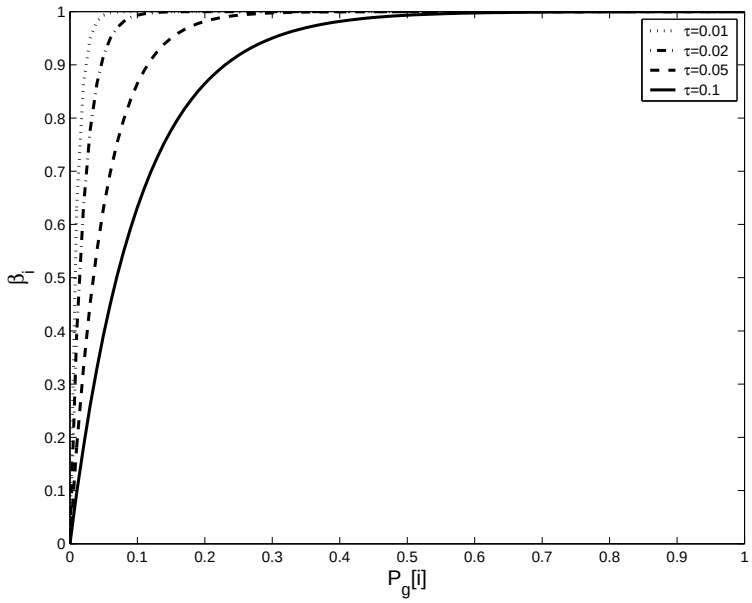


Figure 6.4: The illustration of the function β_i with various τ values.

By varying the tuning factor τ, a designer can adjust the slope of the function β_i. For a larger τ and milder slope of the function β_i, the circuit will consume relatively more power by allowing more glitches. At the same time, it will reduce the number of buffers inserted for the same critical delay requirement. Designers can adjust the value of τ to obtain the desired solution according to their specific needs.

6.3.2 Application to our process-variation-resistant LP model

The input-specific optimization is also applied to our process-variation-resistant LP model proposed in Chapter 5. Under the existences of process variations, our input-specific optimization requires an additional tuning option to avoid undesired solutions.

Modifications to the LP model

Using the concepts of static optimization, selective relaxation and generalized relaxation described previously, the glitch-filtering constraint of our process-variation-resistant LP model (constraint 5.24) is modified as

$$\mu_{d_i} > [\mu_{W_i} + 3 \cdot k(\sigma_{W_i} + r \cdot \mu_{d_i}) \cdot \alpha] \cdot \beta_i; \tag{6.6}$$

The glitch-filtering constraint of gate i is relaxed by β_i. When $\beta_i = 0$, glitch-filtering constraint is altogether removed. Same as before, β_i is a function of $P_g[i]$ and can be chosen from Equation 6.4 or Equation 6.5

Note that, under the existence of process variations, glitches are not always suppressed at every gate. Our adoption of parameter α to relax glitch-filtering constraints may result in some glitches being propagated to the next stage. However, we believe even though glitches can be propagated the next stage, they are more than likely to be suppressed by gates in that stage, the total proportion of these propagated glitches is negligible.

Optional tuning

Under the existence of process variations, the critical delay for a optimized circuit will not be a constant. The delay of critical paths can increases due to the delay variations of gates. Therefore, critical delay of the optimized circuit is a random variable with certain mean and variance. We have found that under process variation, a solution to the input-specific optimization can lead to an undesirable design.

Consider the example shown in Figure 6.5. Under the input-specific optimization, glitch-filtering constraints for all AND/NAND gates are removed because one of the PIs to

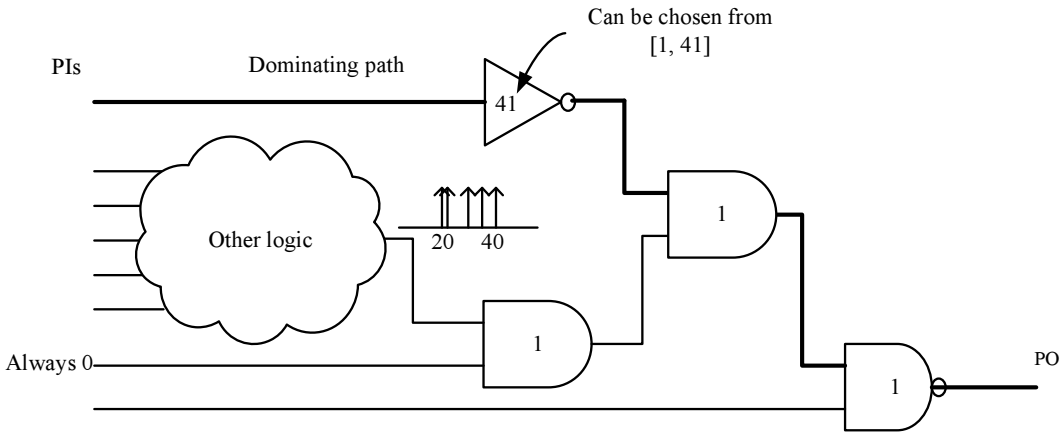


Figure 6.5: An undesired solution under process variations when the input-specific optimization is applied directly. The thick line indicate the dominating path. The number on each gate indicates its inertial delay value.

one AND gate is a constant zero. Delays for these AND/NAND gates are all set to the minimum value, $d_i = 1$. The signal arrival time for the AND gate is between 20 and 40 due to the other logic not shown completely. Under this situation, the delay of the inverter can be chosen anywhere from a minimum value $d_i = 1$ to a maximum value of $d_i = 43 - 2 = 41$. However, in some cases, the LP solver will choose $d_i = 41$ if no constraint prevents it from doing so. This solution is undesired under process variation. The critical path PI, inverter and PO is unnecessary. This path will dominate the critical delay of the circuit under the process-variation and result in the degradation of critical delay distribution.

To avoid this undesirable solution, we consider an additional tuning option to the objective function. The objective of the our LP model was to minimize the sum of buffer delays

$$\texttt{Minimize} \quad \sum_j d_j; \quad (j \in buffers) \tag{6.7}$$

Now, in the input-specific optimization under process-variation, it is replaced by

$$\begin{aligned} \texttt{Minimize} \qquad & \sum_j d_j \; + \; TF \cdot (\tfrac{1}{N} \cdot \sum_i d_i); \\ & (j \in buffers, \; i \in other \; gates) \end{aligned} \tag{6.8}$$

where $TF \geq 0$ is the tuning factor, N is the total number of gates other than buffers.

When $TF > 0$, the tuning option is turned on. The value of TF is less than one so that its impact on the overall optimization is minimized. However, as long as $TF > 0$, there is motivation for the LP solver to minimize those gate delays that do not affect any constraints. With this tuning option, the gates on the dominating paths will be assigned minimum delays.

6.4 Summary

In this chapter, we have introduced the input-specific optimization of a circuit under a given input sequence. An input-specific optimization can reduce the overdesign of the circuit and achieve the same power reduction with less overhead. Our input-specific optimization relies on the static analysis of glitch-generation probabilities and the assumption that no (or only a negligible number of) glitches can be propagated throughout the circuit. Both selective relaxation and generalized relaxation methods are proposed to provide designers more flexibility. The input-specific optimization is first applied to the previous LP model [170] under no process variation. It is then applied to our process-variation-resistant LP model proposed in Chapter 5. Under process variations, an additional tuning option is added to the objective function to eliminate unnecessary delay assignments.

Chapter 7

Experimental Results for Process-Variation-Resistant LP Models

In Chapters 4 and 5, we have proposed LP models based on worst-case timing and statistical timing analyses. In this chapter, we present experimental results obtained from these models. Results are compared with "un-optimized" circuits and "optimized" circuits from the previous LP model [170]. Results for ISCAS'85 benchmark circuits under two different degrees of process-variation are presented. Both power dissipation and critical delay are analyzed.

7.1 Experimental procedure

Our experimental procedure is shown in Figure 7.1. A PERL script is used to extract data from the given circuit. These data include gate numbers, inputs for each gate, single-input-gates, etc. These data are saved in a data file and fed into the AMPL [68] program together with the predefined LP model. Users also need to provide parameters such as, D_{max}, $r = \sigma/\mu$, and optimism factor α. AMPL solves the linear program and gives an optimal solution for all gate delays (nominal value). We use another PERL script to generate an optimized circuit with these delay values. Buffers are inserted as necessary. Since the logic simulator we used only takes delay value as integers, all delays values from AMPL are rounded to the closest integer. Finally, logic simulation is performed to obtain the power dissipation and timing information for the circuit.

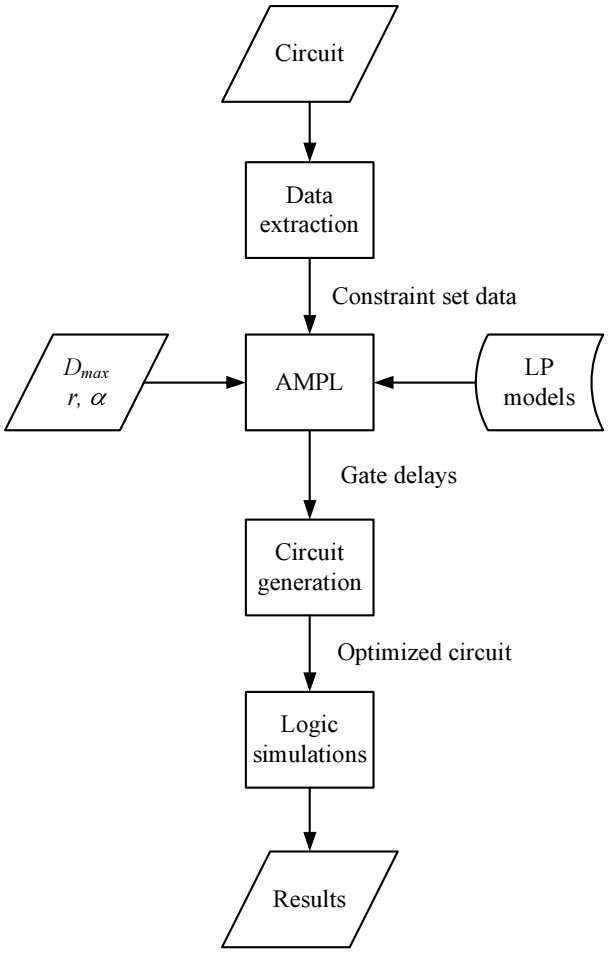


Figure 7.1: The experimental procedure for result analysis.

7.2 Results for small process-variation

In this section, we analyze the optimized circuits from our process-variation-resistant LP models in terms of power dissipation and critical delay under small process-variations. We compare results to "un-optimized" circuits and circuits optimized by a previous approach [170]. We show that our LP models are able to obtain better solutions than the previous approach in [170] under the same circuit delay requirement. Resulting circuits from our LP models are more process-variation-resistant, i.e., they are able to maintain low

power dissipation despite the process-variation. Furthermore, our LP model based on statistical timing is able to suppress the variation of circuit delay. Therefore the critical delay of the optimized circuit has a smaller deviation from its design value under the process-variation. To give more insight into our methods, we first show results for an example circuit with more details. Results for ISCAS'85 benchmark circuit are presented next.

7.2.1 Results for an example circuit

We use the c432 circuit from ISCAS'85 benchmarks as an example. This is the first small and non-trivial ISCAS'85 benchmark circuit. In our experiments, we assume a 5% intra-die delay variation. Therefore, for all gate delay d_i, $r = \sigma/\mu = 0.05$. We assume no inter-die variation. Power estimation is done with 32 stuck-at-fault test vectors (a complete gate level test set).

Power analysis

Power dissipation under no process-variation. The power dissipation under no process-variation is shown in Table 7.1. Power estimation method is the same as that in [170]. Circuits are simulated using an event-driven logic simulator. The signal switching activity data are collected and dynamic power is estimated by the weighted sum of switching activity at the output of every gate. The weights used during the summation are the numbers of fanouts of gates. In essence, we estimate the load capacitance of a gate with the number of fanouts. The average power in Table 7.1 is normalized with respect to the power dissipation of the un-optimized circuit. Same as [170], we use a unit-delay circuit as the un-optimized circuit, where each gate has a delay of one unit.

Un-opt.	Opt [170]			Opt1 (Chp. 4)				Opt2 (Chp. 5)			
Avg. Pwr.	Avg. Pwr.	No. Buf.	Max-delay	Avg. Pwr.	No. Buf.	Param.		Avg. Pwr.	No. Buf.	Param.	
						D_{max}	α			D_{max}	α
1.0	0.74	95	17	0.74	96	20	4.7	0.74	99	20	0.4
1.0	0.74	84	18	0.74	91	21	4	0.74	91	21	0.5
1.0	0.74	80	26	0.74	94	30	2.2	0.74	91	30	0.85
1.0	0.74	66	34	0.74	91	40	1.7	0.74	91	40	1

Table 7.1: Power dissipation under no process-variation and number of buffers inserted for the optimized c432 circuit by various LP models. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively. $r = 0.05$ in "Opt1" and "Opt2".

In Table 7.1, average power, number of buffers inserted, and other parameters are listed for each method. "Opt", "Opt1", and "Opt2" represent the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively. As we can see from the table, all LP models lead to the same power reduction when no process-variation exists. The parameter *maxdelay* in "Opt" represent the maximum delay of the circuit without considering the process-variation, while parameter D_{max} in "Opt1" and "Opt2" represents the maximum delay value under process variation. As will be explained later, D_{max} are chosen accordingly to ensure the same performance.

Note that, when *maxdelay* is allowed to increase from the minimum value of 17, the number of buffers inserted into the circuit by "Opt" reduces. However, in our models, we try to avoid relaxing the constraints too much. Therefore, we adjust α just enough to give a solution. Thus, number of buffers inserted does not necessarily decrease with the increase of D_{max}.

Power dissipation under process-variation. Power dissipation under process-variation (5% intra-die delay variation) is shown in Table 7.2. Monte-Carlo simulation method is used where the optimized circuit is simulated for 1,000 samples of gate delays. That is, after gate

delays are obtained from the linear program, we randomly generate 1,000 samples of gate delays assuming a truncated normal distribution with the given σ/μ ratio. The circuit is simulated for 1,000 delay sample sets to find the distribution of average power and critical delay. In Table 7.2, "Maxdelay" is the maximum delay parameter in [170]; "Mean Pwr." represents the mean of the power distribution; and "Max Dev." represents the difference ratio between the maximum value of the power distribution and the power dissipation under no process-variation. This value shows the deviation of the average power from its design value due to the process-variation. All "Mean Pwr." entries are normalized with respect to power dissipation by the un-optimized circuit under no process-variation.

Max-delay	Un-opt.		Opt		Opt1		Opt2	
	Mean Pwr.	Max. Dev. (%)	Mean Pwr.	Max. Dev.(%)	Mean Pwr.	Max. Dev.(%)	Mean Pwr.	Max. Dev.(%)
17	1.07	17.6	0.78	12.6	0.75	6.8	0.75	4.3
18	1.07	17.6	0.78	13.1	0.75	4.4	0.74	4.4
26	1.07	17.6	0.76	9.1	0.74	0.6	0.74	0.2
34	1.07	17.6	0.76	8.1	0.74	0.1	0.74	0.2

Table 7.2: Power dissipation under 5% intra-die variation for the optimized c432 circuit by various LP models. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively.

We see, when power dissipation of the optimized circuit by previous model ("Opt") increases from its design value under process-variation, our LP models ("Opt1" and "Opt2") are able to suppress such deviation. When the maximum delay of the circuit is allowed to increase, our LP models can eliminate the increase of power dissipation. Obviously, our solutions are resistant to process variations in terms of the power dissipation.

Note that the deviation of power is less for *all* LP models when the maximum delay is allowed to increase. This can be explained considering two major effects that cause the reduction of glitches, *path balancing* and *glitch filtering*. As the maximum delay requirement

is close to the minimum value, path balancing is the major mechanism that reduces glitches. When maximum delay requirement is allow to increase, more glitches are removed by the increase of gate inertial delays. Under process-variation, path balancing is more sensitive to the change of delays and results in a larger increase of power dissipation.

Delay analysis

The critical delays of circuits under process-variation are shown in Table 7.3. "Maxdelay" is the maximum delay parameter [170] (maximum allowed critical path delay); "Nom. Delay" indicates the critical delay of the circuit under no process-variation, the nominal value of the critical path delay. We show the *mean* and *std. dev.* of the distribution of critical delay under the process-variation. We also show the maximum deviation ("Max. Dev.") of the critical delay from its intended value.

As mention earlier, we choose D_{max} for our models considering the existence of process variation. As we can see from Table 7.3(a), under the process variation, the critical delay of the optimized circuit "Opt" deviates from its nominal value. In the worst-case, it is $1 + 3 \cdot \sigma/\mu$ times of the nominal value. In order to ensure the same circuit performance under the process variation, we choose $D_{max} = (1+3 \cdot \sigma/\mu) \cdot maxdelay$. As we see from table, by choosing D_{max} this way, the resulting circuits actually have same critical delay under no process-variation. In most case, "Opt1" and "Opt2" have a similar delay distribution (in terms of mean and std. dev.) to "Opt" under the process-variation. However, when $maxdelay = 34$, "Opt2" is able to suppress the maximum deviation of critical delay from 11.6% to 7.2% and thus has the best delay performance. Note that all optimized circuits are affected by quantization errors of gate delays. The nominal delay of the optimized circuit may not satisfy the original $maxdelay$ constraint precisely.

Max-delay	Un-opt.				Opt			
	Nom. Delay	Mean	Std. Dev.	Max. Dev. (%)	Nom. Delay	Mean	Std. Dev.	Max. Dev. (%)
17	17	17.45	0.18	5.8	17	18.09	0.26	11.0
18	17	17.45	0.18	5.8	19	20.00	0.24	9.1
26	17	17.45	0.18	5.8	27	28.05	0.4	8.3
34	17	17.45	0.18	5.8	34	35.95	0.67	11.6

(a)

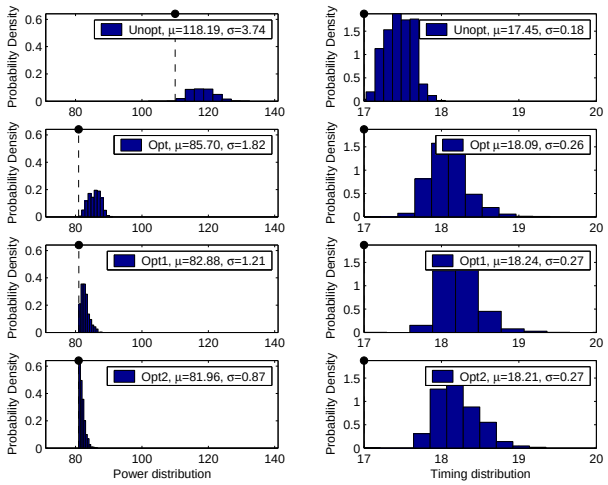
Max-delay	Opt1				Opt2			
	Nom. Delay	Mean	Std. Dev.	Max. Dev. (%)	Nom. Delay	Mean	Std. Dev.	Max. Dev. (%)
17	17	18.24	0.27	12.1	17	18.21	0.27	11.9
18	19	20.04	0.29	10.1	19	19.99	0.3	9.9
26	27	28.66	0.38	10.4	27	27.83	0.4	7.5
34	35	37.05	0.58	10.8	35	36.05	0.49	7.2

(b)

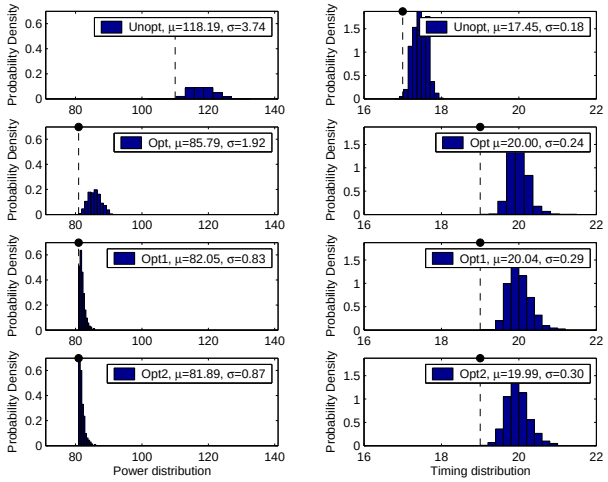
Table 7.3: Critical delay distributions under 5% intra-die variation for the optimized c432 circuit by various LP models. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively.

Power-delay analysis

The above analysis investigated the power dissipation and critical delay separately. Here we show the distribution of both in Figures 7.2 and 7.3. We can see more clearly, the distribution of power dissipation by "Opt1" and "Opt2" is much sharper than that by "Opt". By allowing the increase of circuit critical delay, "Opt1" and "Opt2" almost completely eliminate the variation of power under the process variation. We also observe that critical delays for all optimized circuits are more sensitive to the process variation than the un-optimized circuits. This is because many paths in an optimized circuit are balanced and critical while there are only few critical paths in the un-optimized circuit. Under the process-variation, the critical delay of an optimized circuit is more prone to increase than in

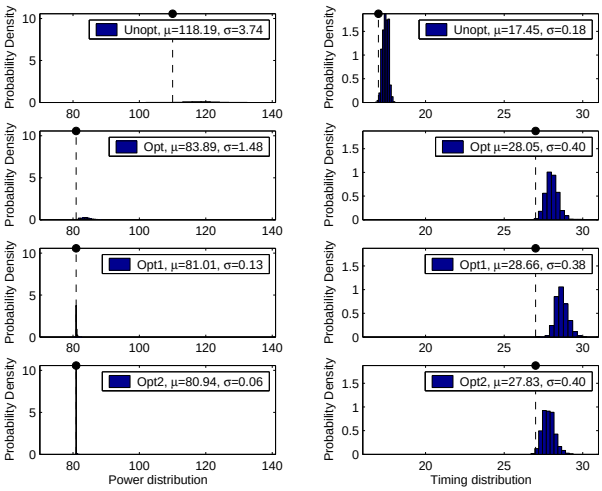


(a)

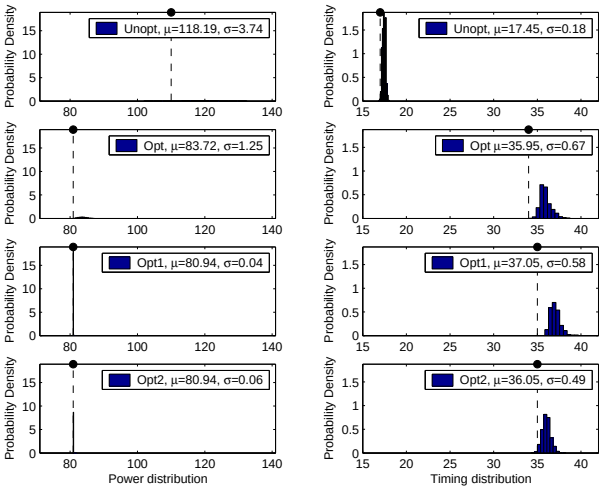


(b)

Figure 7.2: Power and timing distributions under 5% intra-die variation for the c432 circuit: (a) power and timing distribution when $maxdelay = 17$, (b) power and timing distribution when $maxdelay = 18$. For each figure, the left column shows the distributions of power dissipation of the circuit. The X-axis represents the power dissipation (not normalized) and Y-axis represents the probability density. For each figure, the right column shows the distributions of critical delay. The X-axis represents the time and Y-axis represents the probability density. The nominal value under no process-variation is plotted in dashed line with a solid circle at the top. "Un-opt" represents the un-optimized circuit. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively.



(a)



(b)

Figure 7.3: Power and timing distributions under 5% intra-die variation for the c432 circuit: (a) power and timing distribution when $maxdelay = 26$, (b) power and timing distribution when $maxdelay = 34$. For each figure, the left column shows the distributions of power dissipation of the circuit. The X-axis represents the power dissipation (not normalized) and Y-axis represents the probability density. For each figure, the right column shows the distributions of critical delay. The X-axis represents the time and Y-axis represents the probability density. The nominal value under no process-variation is plotted in dashed line with a solid circle at the top. "Un-opt" represents the un-optimized circuit. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively.

the un-optimized circuit. In most cases, optimized circuits "Opt", "Opt1" and "Opt2" have similar distributions of critical delay under the process-variation. While in Figure 7.3(b), we see "Opt2" is able to suppress the deviation of critical delay and therefore is more resistant to the process-variation than are "Opt" and "Opt1".

We plot the relationship between power dissipation and critical delay in Figure 7.4. We can see that when critical delay of the circuit is allowed to increase, the power distribution with a lower mean and maximum value can be obtained. Also, the difference between mean and maximum values are reduced, which means that the circuit is more resistant to the process variation. Any point along the curve indicates a possible solution and users can choose a design according to their requirements.

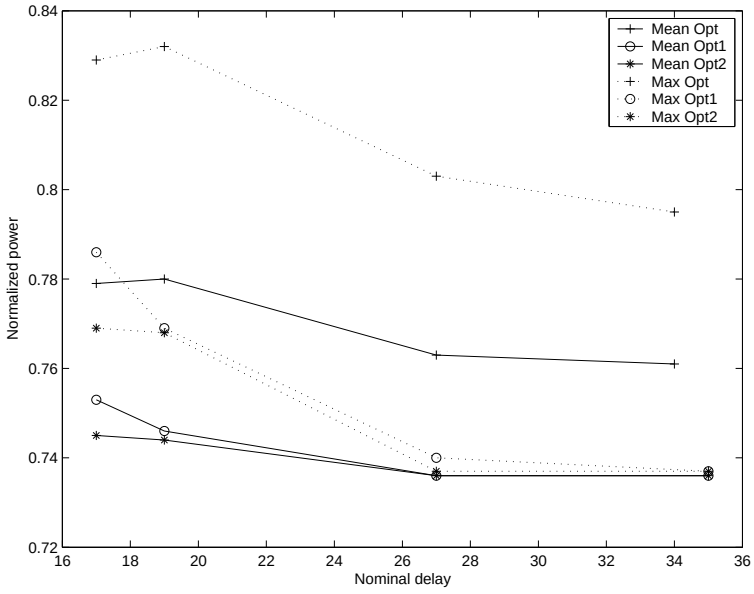


Figure 7.4: Relationship between average power (mean and maximum value) and critical delay under 5% intra-die variation for the optimized c432 circuit by different LP models. The X-axis represents the nominal critical delay of the circuit under no process-variation. The Y-axis represents the normalized power value.

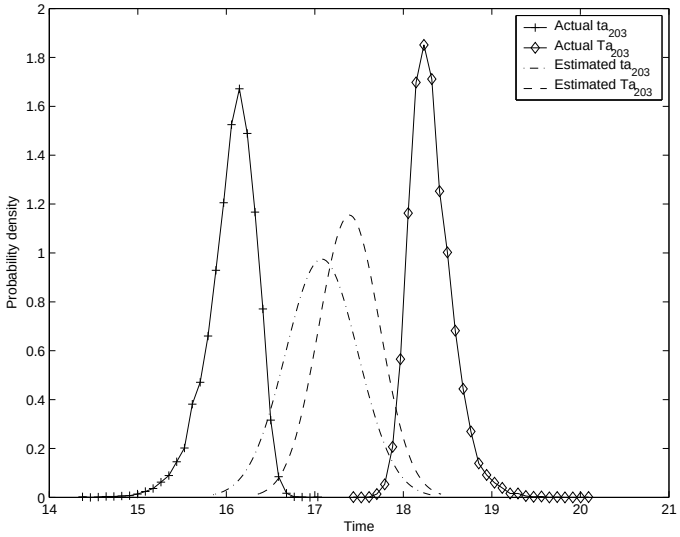
Timing accuracy

Finally, we investigate the accuracy of our statistical timing analysis and degree of errors introduced by our approximations. As discussed in Chapter 5, the approximations could contribute to errors in the estimation of minimum and maximum signal arrival times. To verify the accuracy of our timing analysis, we analyze the distribution of minimum and maximum signal arrival times at the primary output (gate 203) of c432 circuit. The Monte-Carlo analysis method is used where 10,000 samples of gate delays are generated. Optimized circuit by "Opt2" is simulated with those 10,000 samples of gate delays to obtain the statistics of the signal arrival times. Figure 7.5 shows the estimated ta_{203} and Ta_{203}, and actual ta_{203} and Ta_{203} from Monte-Carlo analysis for $D_{max} = 20$ and $D_{max} = 40$.

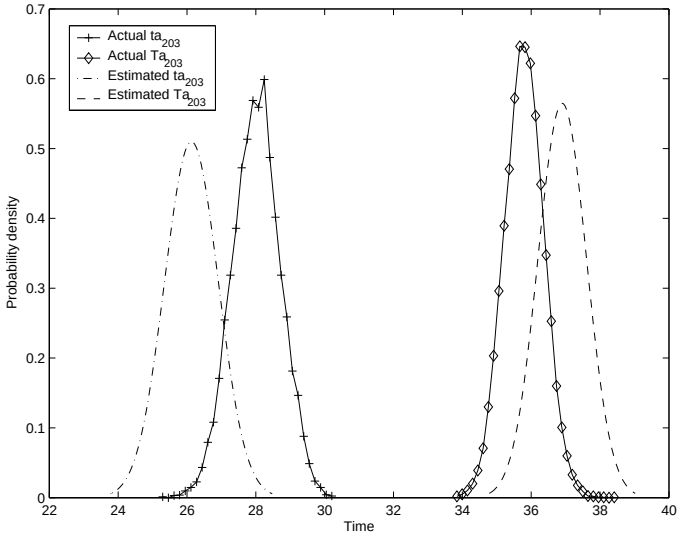
We see our estimation leads to a larger error when D_{max} is smaller. When D_{max} is large, the estimation has a better match to the actual distribution. This is because when D_{max} is small, paths are more balanced in the optimized circuit and the means of signal arrival times are close together. As shown in Figure 5.2, this is the case in which our approximation leads to a larger error. The estimated and actual ta_{203} and Ta_{203} for circuits optimized by "Opt2" at different D_{max} are shown in Table 7.4.

	ta_{203}						**Ta_{203}**					
	Actual		**Est.**		**Err. (%)**		**Actual**		**Est.**		**Err. (%)**	
D_{max}	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
20	16.06	0.27	17.08	0.35	6.3	26	18.31	0.24	17.39	0.41	-5.0	70
21	17.23	0.27	16.93	0.38	-1.8	41	19.77	0.24	19.46	0.40	-1.6	68
30	23.03	0.47	21.75	0.61	-5.6	31	26.85	0.33	27.43	0.45	2.2	34
40	28.02	0.69	26.13	0.78	-6.7	13	35.79	0.57	36.90	0.71	3.1	24

Table 7.4: The accuracy of our statistical timing analysis. The estimated value and actual timing statistics for ta_{203}, Ta_{203} are compared for c432 circuit optimized by "Opt2" at different D_{max}.



(a)



(b)

Figure 7.5: The probability distribution for estimated ta_{203}, Ta_{203} and actual ta_{203}, Ta_{203} for c432 circuit optimized by "Opt2": (a) the estimated ta_{203}, Ta_{203} and actual ta_{203}, Ta_{203} for $D_{max} = 20$, (b) the estimated ta_{203}, Ta_{203} and actual ta_{203}, Ta_{203} for $D_{max} = 40$.

From Table 7.4, we see that the estimation error in the mean value is less than 7%. For a large D_{max}, the estimation error in σ is less than 25%.

7.2.2 Results for ISCAS'85 benchmark circuits

ISCAS'85 benchmark circuits are optimized using our LP models to show the effectiveness of our models. In our optimizations, 5% intra-die variation is used ($r = 0.05$ for "Opt1" and "Opt2"). Power estimations for smaller circuits (i.e., c432, c499, c880, and c1355) are done using complete (100% stuck fault coverage) gate level test vectors. For larger circuits, 50 random vectors with signal probability of 0.5 are used.

Power analysis

Power dissipation under no process-variation. The power dissipation under no process-variation is shown in Table 7.5.

We see, in most cases, circuits optimized by "Opt", "Opt1", and "Opt2" have the same power reduction as when no process-variation exists. As mentioned before, D_{max} are chosen according to *maxdelay* to ensure the same delay performance. For some large circuits, e.g., c5315 and c7552, our LP model "Opt1" failed to give a good optimization if maximum delay requirement was not allowed to increase from its minimum value. This is because "Opt1" is based on the worst-case timing analysis. When a circuit has more levels, the worst-case timing analysis tends to be very pessimistic. We have to adjust the optimism factor α dramatically to obtain the solution and we may not achieve good optimization. We observe that in most cases, especially when the maximum delay requirement is allowed to increase, "Opt1" and "Opt2" insert more buffers to the circuit in order to obtain a process-variation-resistant design.

Circuit	Un-opt.	Opt			Opt1			Opt2		
	Avg. Pwr.	Avg. Pwr.	No. Buf.	Max-delay	Avg. Pwr.	No. Buf.	D_{max}	Avg. Pwr.	No. Buf.	D_{max}
c432	1.0	0.74	95	17	0.74	96	20	0.74	99	20
	1.0	0.74	66	34	0.74	91	40	0.74	91	40
c499	1.0	0.94	80	11	0.94	88	13	0.94	97	13
	1.0	0.94	48	22	0.94	88	26	0.94	129	26
c880	1.0	0.54	63	24	0.54	45	28	0.54	76	28
	1.0	0.54	29	72	0.54	37	83	0.54	37	83
c1355	1.0	0.93	224	24	0.93	296	28	0.93	305	28
	1.0	0.93	160	72	0.93	296	83	0.93	273	83
c1908	1.0	0.53	84	40	0.53	68	46	0.52	136	46
	1.0	0.55	54	120	0.53	92	138	0.52	198	138
c2670	1.0	0.74	157	32	0.79	244	37	0.73	313	37
	1.0	0.74	26	96	0.75	80	111	0.73	168	111
c3540	1.0	0.60	219	47	0.59	228	55	0.59	306	55
	1.0	0.59	103	141	0.61	152	163	0.59	303	163
c5315	1.0	0.56	281	49	0.62	228	57	0.55	401	57
	1.0	0.56	113	147	0.58	130	170	0.55	460	170
c6288	1.0	0.13	881	124	0.15	801	143	0.14	1685	143
	1.0	0.13	864	372	0.14	922	428	0.13	1213	428
c7552	1.0	0.52	369	43	0.64	180	50	0.52	464	50
	1.0	0.52	62	129	0.56	162	149	0.52	879	149

Table 7.5: Power dissipation with no process-variation and number of buffers inserted for the optimized ISCAS'85 benchmark circuits by various LP models. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively. $r = 0.05$ in "Opt1" and "Opt2".

Power dissipation under process-variation. The power dissipation under the process-variation is shown in Table 7.6. Unlike the c432 example circuit, here we apply a 5% intra-die variation and a 5% inter-die variation.

Comparing to "Opt", our optimization methods "Opt1" and "Opt2" can further reduce the mean and the deviation (increase) of power dissipation under the process variation. While "Opt1" is not able to reduce the mean of power dissipation for certain large circuits, e.g., c5315 and c7552, our LP model based on statistical timing "Opt2", always obtains a better solution that is more resistant to the process-variation. It also shows that for

Circuit	Max-delay	Un-opt. Mean Pwr.	Un-opt. Max. Dev. (%)	Opt Mean Pwr.	Opt Max. Dev. (%)	Opt1 Mean Pwr.	Opt1 Max. Dev. (%)	Opt2 Mean Pwr.	Opt2 Max. Dev. (%)
c432	17	1.08	17.5	0.78	12.8	0.75	7.0	0.75	4.5
	34	1.08	17.5	0.76	8.2	0.74	0.1	0.74	0.1
c499	11	1.06	12.9	1.00	12.6	0.95	0.7	0.95	0.7
	22	1.06	12.9	0.99	12.6	0.94	0.0	0.94	0.1
c880	24	1.03	7.1	0.62	23.1	0.58	13.9	0.55	7.5
	72	1.03	7.1	0.57	12.8	0.55	1.1	0.54	1.0
c1355	24	1.10	18.1	0.99	10.6	0.96	5.5	0.95	4.2
	72	1.10	18.1	0.98	8.8	0.93	0.3	0.93	0.1
c1908	40	1.15	21.0	0.64	28.6	0.62	22.8	0.58	21.6
	120	1.15	21.0	0.64	21.5	0.54	5.9	0.54	6.5
c2670	32	1.17	21.8	0.80	11.6	0.81	5.5	0.75	4.8
	96	1.17	21.8	0.77	6.1	0.78	5.2	0.74	1.8
c3540	47	1.15	18.9	0.66	15.2	0.65	12.9	0.63	9.7
	141	1.15	18.9	0.62	7.2	0.63	5.1	0.59	1.3
c5315	49	1.12	14.9	0.62	13.8	0.67	9.9	0.59	9.1
	147	1.12	14.9	0.60	10.3	0.61	6.8	0.56	3.7
c6288	124	1.46	49.9	0.27	131.6	0.28	105.9	0.24	93.6
	372	1.46	49.9	0.26	128.3	0.23	76.8	0.18	56.0
c7552	43	1.17	19.6	0.57	12.4	0.72	13.3	0.57	11.8
	129	1.17	19.6	0.56	9.3	0.58	5.1	0.53	3.5

Table 7.6: Power dissipation with 5% inter-die variation and 5% intra-die variation for the optimized ISCAS'85 benchmark circuits by various LP models. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively.

large circuits, e.g., c5315, c6288 and c7552, optimization is more difficult. Only limited reduction in the mean and deviation of power dissipation are obtained when circuit delay is not allowed to increase. Since there are more levels of logic in these large circuits, one may have to sacrifice more circuit performance in order to suppress the deviation of power. Finally, as we mentioned before, power dissipation of a circuit is not affected by the inter-die variation. This can be see by comparing the power dissipation for c432 circuit with that in Table 7.2.

Delay analysis

The critical delay distributions under the process-variation (5% intra-die variation and 5% inter-die variation) are shown in Figure 7.6. Note that, even though inter-die variation has negligible effect on the power dissipation, it does affects the circuit delay. As we can see, the maximum deviation of critical delay under the process-variation is now more than 15%.

In most cases, "Opt", "Opt1" and "Opt2" maintain similar performances in terms of the nominal delay under no process-variation and the mean value of the delay distribution. From Figure 7.6(d), we can clearly observe the maximum deviation of critical delay from its nominal value. Under small process-variation, we do not observe an overwhelming trend of the reduction of delay deviation by our optimization method "Opt2". However, we can see in some cases, e.g., c880, c2670, c5315 and c7552, the optimization by "Opt2" reduced the maximum deviation of critical delay to a smaller value.

7.3 Results for large process-variation

In this section, we analyze resulting circuits from our LP models for power dissipation and critical delay under a relatively large process-variation. We show that our LP models are still able to obtain better solutions than the previous approach [170] under a large process-variation. Resulting circuits from our LP models are more process-variation-resistant in terms of both power and delay performance. To give a complete analysis of our LP models, we first show results for a large example circuit with more details and then we present results for ISCAS'85 benchmark circuits.

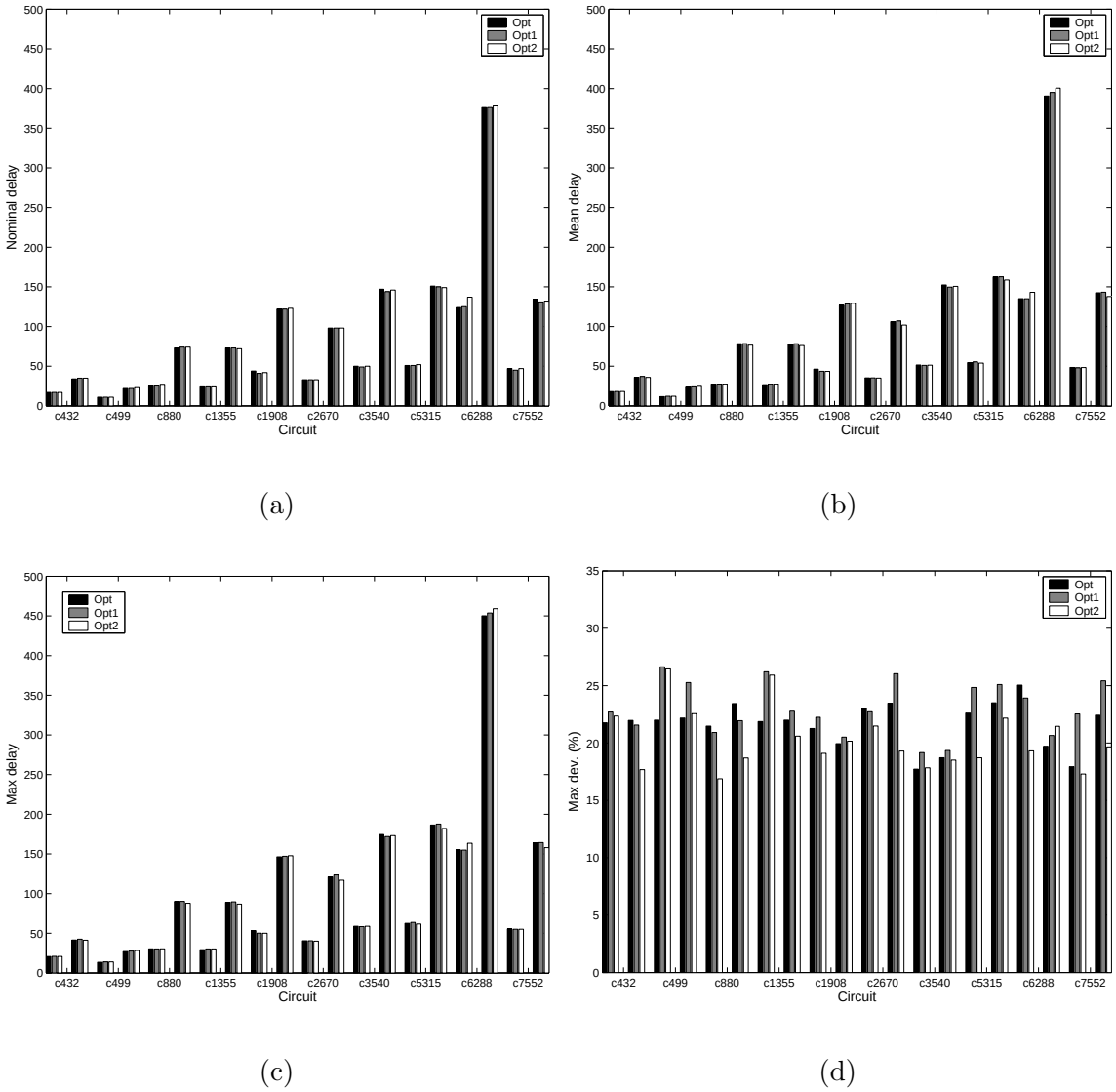


Figure 7.6: Critical delay for the optimized ISCAS'85 benchmark circuits under 5% inter-die and 5% intra-die delay variation by various LP models: (a) the nominal critical delays under no process-variation, (b) the mean values of critical delay under the process variation, (c) the maximum values of critical delay under the process variation, (d) the maximum deviation of critical delay (in percentage) from the nominal delay under the process variation. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively. For each circuit, delay results for two different *maxdelay* parameters are shown.

7.3.1 Results for an example circuit

To show how our LP models work for a large circuit, we use the c7552 circuit from ISCAS'85 benchmarks as an example, which is the largest circuit in ISCAS'85 benchmarks. In our experiments, we assume a 15% intra-die delay variation and a 5% inter-die variation ($r = 0.15$ for "Opt1" and "Opt2"). Power estimation is done with 50 random vectors with the signal probability of 0.5.

Power analysis

Power dissipation under no process-variation. The power dissipation under no process-variation is shown in Table 7.7. The average power in Table 7.7 is normalized with respect to the power dissipation of the un-optimized circuit. Same as in the previous section, we use a unit-delay circuit as the un-optimized circuit.

Un-opt.	**Opt**			**Opt1**				**Opt2**			
Avg. Pwr.	**Avg. Pwr.**	**No. Buf.**	**Max-delay**	**Avg. Pwr.**	**No. Buf.**	**Param.**		**Avg. Pwr.**	**No. Buf.**	**Param.**	
						$\mathbf{D_{max}}$	α			$\mathbf{D_{max}}$	α
1.0	0.52	369	43	0.72	163	63	58	0.52	591	63	0.09
1.0	0.52	91	86	0.69	64	125	21	0.52	481	125	0.22
1.0	0.52	62	129	0.65	87	187	16	0.52	511	187	0.28
1.0	0.52	44	215	0.60	622	312	11	0.52	645	312	0.35

Table 7.7: Power dissipation under no process-variation and number of buffers inserted for the optimized c7552 circuit by various LP models. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively. $r = 0.15$ in "Opt1" and "Opt2".

We can see from Table 7.7 that under a large process-variation, it is more difficult to optimize. A much larger optimism factor α has to be adopted for "Opt1" to obtain a solution. With such a large α, "Opt1" fails to do a good optimization and results in a more power-consuming circuit even if no process-variation exists. However, our LP model

based on statistical timing, "Opt2", is able to optimize the circuit and give a glitch-free circuit (under no process-variation). In general, "Opt2" requires more buffers to obtain a process-variation-resistant circuit.

Power dissipation under process-variation. Power dissipation under process-variation (15% intra-die and 5% inter-die delay variation) is shown in Table 7.8. Monte-Carlo simulation method is used where the optimized circuit is simulated for 1,000 samples of gate delays. In Table 7.8, "Maxdelay" is the maximum delay parameter [170]; "Mean Pwr." represents the mean of the power distribution; and "Max Dev." represents the difference ratio between the maximum value of the power distribution and the power dissipation under no process-variation. All "Mean Pwr." are normalized with respect to power dissipation of the un-optimized circuit under no process-variation.

	Un-opt.		**Opt**		**Opt1**		**Opt2**	
Max-delay	**Mean Pwr.**	**Max. Dev. (%)**	**Mean Pwr.**	**Max. Dev.(%)**	**Mean Pwr.**	**Max. Dev.(%)**	**Mean Pwr.**	**Max. Dev.(%)**
43	1.17	21.9	0.66	32.7	0.87	24.8	0.67	37.4
86	1.17	21.9	0.64	25.8	0.78	16.0	0.59	18.7
129	1.17	21.9	0.61	20.5	0.71	12.3	0.57	15.2
215	1.17	21.9	0.60	20.2	0.65	11.2	0.56	11.8

Table 7.8: Power dissipation under 15% intra-die variation and 5% inter-die variation for c7552 circuit by various LP models. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively.

Under larger process-variation, power dissipation increases more from its expected value. The maximum deviation of power dissipation by "Opt" can be as large as 32.7%. For this large circuit, our LP model "Opt1" failed to do a good optimization. However, the LP model "Opt2" is able to reduce both mean and maximum deviation of power dissipation as compared to "Opt", when the circuit delay is allowed to increase. According to Theorem

2 (Chapter 4), a process-variation-resistant solution cannot be obtained without increasing the overall circuit delay in this case. When the maximum delay increases, our LP models are able to give improved solutions in terms of power dissipation.

Delay analysis

The critical delays of circuits under process-variation are shown in Table 7.9. "Maxdelay" is the specified maximum delay parameter [170]; "Nom. Delay" indicate the critical delay of the circuit under no process-variation, the nominal value of the critical delay. We show that the *mean* and *std. dev.* of the distribution of critical delay under the process-variation. We also show the maximum deviation ("Max. Dev.") of the critical delay from its design value.

As we see from Table 7.9(a), under a large process variation, the critical delay of the optimized circuit "Opt" deviates much more from its nominal value. The maximum deviation of this critical delay can be as large as 55.4%. From Table 7.9(b), we see that "Opt1" does not improve the deviation in critical delay. However, "Opt2" has a smaller deviation (increase) of critical delay under process-variation. The maximum deviation of critical delay here is no more than 35.5%.

The above phenomena can be explained on the basis of the underlying timing model for "Opt1" and "Opt2". For "Opt1", where the worst-case timing analysis is adopted, the LP model makes no attempt to minimize the delay variations. However, for "Opt2", we used the statistical timing analysis. The glitch-filtering constraint ensures that the random variable $d_i - (Tb_i - tb_i)$ is greater than zero. During the optimization, the LP solver tries to minimize both the mean and the variance of the random variable $Tb_i - tb_i$. As the result, the total variance of the critical delay is reduced.

Max-delay	Un-opt.				Opt			
	Nom. Delay	Mean	Std. Dev.	Max. Dev. (%)	Nom. Delay	Mean	Std. Dev.	Max. Dev. (%)
43	43	44.1	2.3	18.7	47	57.6	3.6	45.5
86	43	44.1	2.3	18.7	89	115.1	7.1	53.2
129	43	44.1	2.3	18.7	135	172.2	10.9	51.7
215	43	44.1	2.3	18.7	220	287.2	18.2	55.4

(a)

Max-delay	Opt1				Opt2			
	Nom. Delay	Mean	Std. Dev.	Max. Dev. (%)	Nom. Delay	Mean	Std. Dev.	Max. Dev. (%)
43	44	57.5	3.7	55.7	46	53.4	2.6	33.1
86	87	114.6	7.6	57.8	90	103.7	5.1	32.1
129	131	172.7	10.7	56.3	131	154.8	7.6	35.5
215	221	286.9	17.9	54.1	218	256.8	12.9	35.5

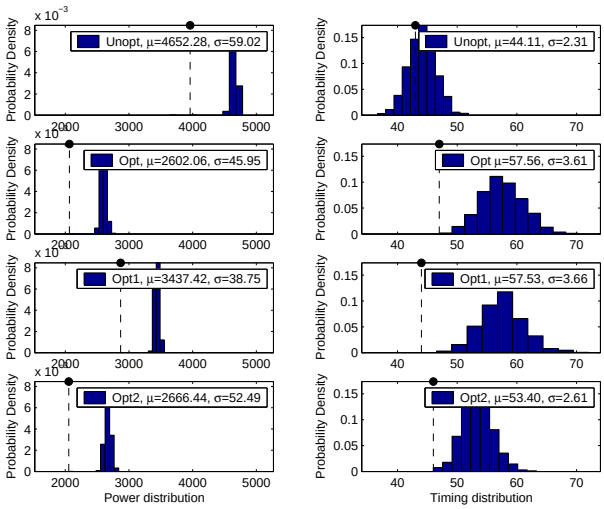
(b)

Table 7.9: Critical delay distributions under 15% intra-die variation and 5% inter-die variation for the optimized c7552 circuit by various LP models. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively.

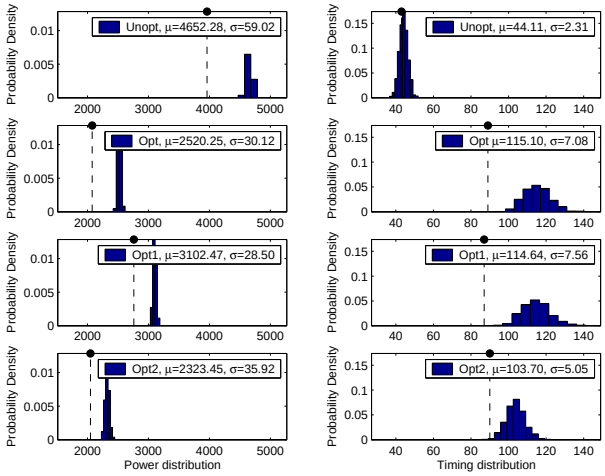
Power-delay analysis

We show the distribution of both power and critical delay in Figures 7.7 and 7.8. We observe that, while it is more difficult to optimize a large circuit under large process variation, the deviation of power dissipation with "Opt2" is smaller than that by "Opt" when the maximum delay specification is allowed to increase. For all cases, the distribution of critical delay by "Opt2" is sharper than those of "Opt" and "Opt1" and therefore "Opt2" is also more process-variation-resistant in terms of the critical delay. By allowing an increase in the circuit delay, "Opt1" and "Opt2" can obtain still better solutions.

We plot the relationship between power dissipation and circuit delay in Figure 7.9. We see that for a large circuit under a large process-variation, "Opt1" does not do a good

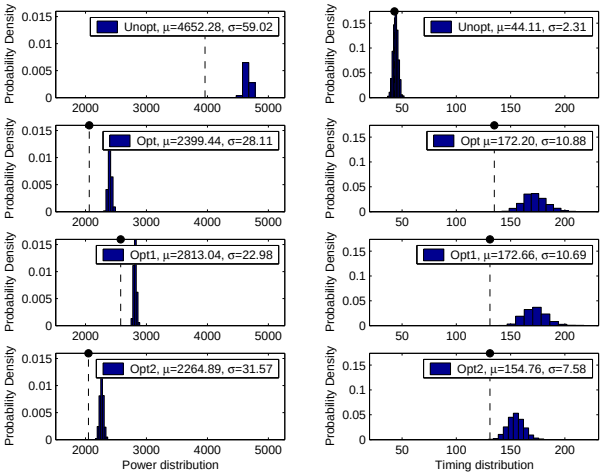


(a)

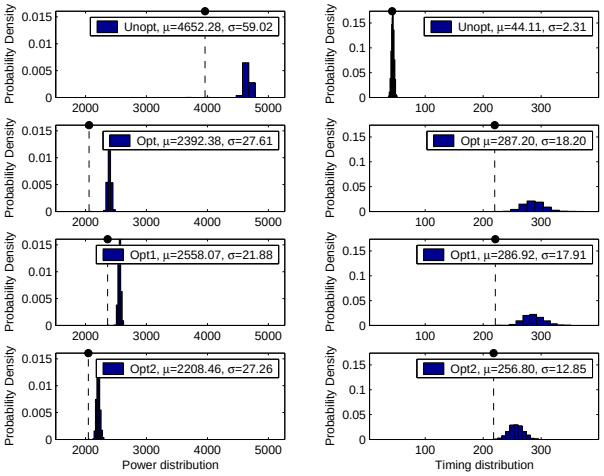


(b)

Figure 7.7: Power and timing distributions under 15% intra-die variation and 5% inter-die variation for the c7552 circuit: (a) power and timing distribution when $maxdelay = 43$, (b) power and timing distribution when $maxdelay = 86$. For each figure, the left column shows the distributions of power dissipation of the circuit. The X-axis represents the power dissipation (not normalized) and Y-axis represents the probability density. For each figure, the right column shows the distributions of critical delay. The X-axis represents the time and Y-axis represents the probability density. The nominal value under no process-variation is plotted in dashed line with a solid circle at the top. "Un-opt" represents the un-optimized circuit. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively.



(a)



(b)

Figure 7.8: Power and timing distributions under 15% intra-die variation and 5% inter-die variation for the c7552 circuit: (a) power and timing distribution when $maxdelay = 129$, (b) power and timing distribution when $maxdelay = 215$. For each figure, the left column shows the distributions of power dissipation of the circuit. The X-axis represents the power dissipation (not normalized) and Y-axis represents the probability density. For each figure, the right column shows the distributions of critical delay. The X-axis represents the time and Y-axis represents the probability density. The nominal value under no process-variation is plotted in dashed line with a solid circle at the top. "Un-opt" represents the un-optimized circuit. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively.

optimization in terms of mean and maximum power dissipation. When critical delay of the circuit is allowed to increase, "Opt2" can obtain a power distribution with a lowest mean and maximum value. In addition, the differences between mean and maximum values are reduced, which indicates that the solution is more resistant to process-variation. Any point along the connected data indicates a possible solution, showing the trade off between the power (variations) and the circuit speed.

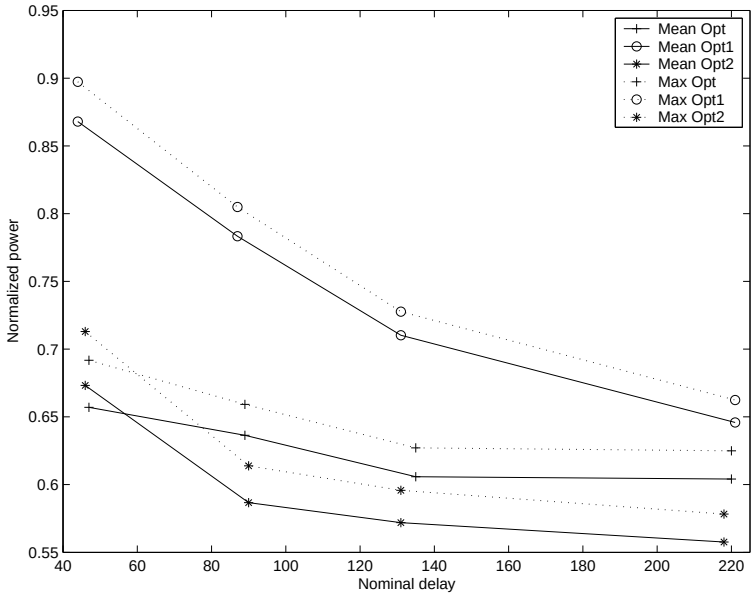


Figure 7.9: Relationship between average power (mean and maximum value) and critical delay under 15% intra-die variation and 5% inter-die variation for the optimized c7552 circuit by different LP models. The X-axis represents the nominal critical delay of the circuit under no process-variation. The Y-axis represents the normalized power value.

7.3.2 Results for ISCAS'85 benchmark circuits

ISCAS'85 benchmark circuits were optimized using our LP models under a large process variation. In this optimization, 15% intra-die variation and 5% inter-die variation were

assumed ($r = 0.15$ for "Opt1" and "Opt2"). Same sets of vectors as in the previous section were used for power estimation. For completeness, results for c7552 circuit are repeated.

Power dissipation under no process-variation

The power dissipation under no process-variation is shown in Table 7.10. In most cases, circuits optimized by "Opt" and "Opt2" have the same power reduction when no process-variation is assumed. That means the solution by "Opt2" is a glitch-free circuit under no process-variation. However, for such a large process-variation assumed, "Opt1" does not give a totally glitch-free circuit even when no process-variation exists. The power dissipation by "Opt1" is always larger than that by "Opt", especially for some large circuits, e.g., c2670, c3540 and c5315. In most cases, the optimization by "Opt1" and "Opt2" requires more buffers to ensure that the resulting circuit is process-variation-resistant.

Power dissipation under process-variation

The power dissipations in the presence of process-variation is shown in Table 7.11. As mentioned above, we apply a 15% intra-die variation and a 5% inter-die variation in these experiments. We see that even under such large process-variation, our optimizations by "Opt1" and "Opt2" still improve (reduce) the mean and deviation of power distribution as compared to "Opt". While "Opt1" failed to do a good optimization for certain large circuits (especially when critical delay was not allowed to increase), e.g., c2670, c3540, and c5315, our LP model based on statistical timing analysis, "Opt2", was able to reduce the mean and deviation of power distribution for all circuits. Better results were obtained by both "Opt1" and "Opt2" when critical delay specification was allowed to increase. Table 7.11 also shows the optimization for larger process-variation is more difficult because path delays

Circuit	Un-opt.	Opt			Opt1			Opt2		
	Avg. Pwr.	Avg. Pwr.	No. Buf.	Max-delay	Avg. Pwr.	No. Buf.	D_{max}	Avg. Pwr.	No. Buf.	D_{max}
c432	1.00	0.74	66	34	0.75	87	50	0.74	88	50
	1.00	0.74	58	68	0.74	81	99	0.74	106	99
c499	1.00	0.94	48	22	0.97	88	32	0.94	88	32
	1.00	0.94	0	33	0.97	0	48	0.94	129	48
c880	1.00	0.54	35	48	0.58	36	70	0.54	57	70
	1.00	0.54	30	120	0.59	29	174	0.54	62	174
c1355	1.00	0.93	192	48	0.95	264	70	0.93	305	70
	1.00	0.93	128	120	0.96	264	174	0.93	305	174
c1908	1.00	0.53	62	80	0.55	41	116	0.52	135	116
	1.00	0.54	34	200	0.56	12	290	0.52	190	290
c2670	1.00	0.74	34	64	0.80	39	93	0.74	249	93
	1.00	0.74	9	160	0.78	95	232	0.73	211	232
c3540	1.00	0.59	139	94	0.62	149	137	0.59	281	137
	1.00	0.59	78	235	0.65	52	341	0.59	311	341
c5313	1.00	0.56	167	98	0.66	93	143	0.55	399	143
	1.00	0.56	53	245	0.60	144	356	0.55	418	356
c6288	1.00	0.13	870	228	0.14	1303	331	0.13	1121	331
	1.00	0.13	857	620	0.13	939	899	0.13	1473	899
c7552	1.00	0.52	91	86	0.69	64	125	0.52	481	125
	1.00	0.52	44	215	0.60	622	312	0.52	645	312

Table 7.10: Power dissipation under no process-variation and number of inserted buffers for optimized ISCAS'85 benchmark circuits by various LP models. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively. $r = 0.15$ in "Opt1" and "Opt2".

will have larger variations. One may have to sacrifice more circuit performance (let delay increase) to obtain a process-variation-resistant design.

Delay analysis

The critical delay distribution in the presence of process-variation is shown in Figure 7.10. We see that "Opt", "Opt1" and "Opt2" maintain a similar performance in terms of the nominal delay. From the mean value, maximum value, and maximum deviation of

Circuit	Max-delay	Un-opt.		Opt		Opt1		Opt2	
		Mean Pwr.	Max. Dev. (%)	Mean Pwr.	Max. Dev. (%)	Mean Pwr.	Max. Dev. (%)	Mean Pwr.	Max. Dev. (%)
c432	34	1.09	19.8	0.78	12.6	0.78	12.1	0.76	11.1
	68	1.09	19.8	0.77	10.3	0.75	6.1	0.74	3.7
c499	22	1.07	14.0	1.02	15.3	0.98	1.7	0.95	2.0
	33	1.07	14.0	0.99	10.2	0.97	1.4	0.95	1.0
c880	48	1.04	8.0	0.62	26.5	0.63	15.7	0.59	18.2
	120	1.04	8.0	0.60	22.7	0.60	5.6	0.55	8.6
c1355	48	1.13	21.8	1.06	19.7	0.98	7.3	0.98	10.2
	120	1.13	21.8	1.05	18.8	0.97	1.7	0.94	3.0
c1908	80	1.16	23.1	0.72	49.6	0.66	30.1	0.64	35.8
	200	1.16	23.1	0.66	32.3	0.62	18.8	0.58	21.4
c2670	64	1.19	25.4	0.81	13.6	0.90	16.0	0.80	13.6
	160	1.19	25.4	0.80	11.2	0.82	8.6	0.76	6.2
c3540	94	1.16	20.7	0.67	19.5	0.69	16.9	0.66	17.8
	235	1.16	20.7	0.66	16.1	0.71	11.7	0.62	10.1
c5313	98	1.13	16.5	0.67	24.6	0.74	16.3	0.63	20.8
	245	1.13	16.5	0.64	19.0	0.66	13.9	0.60	13.4
c6288	228	1.45	52.2	0.43	274.3	0.36	193.4	0.38	223.8
	620	1.45	52.2	0.41	264.0	0.31	161.5	0.26	125.3
c7552	86	1.17	21.9	0.64	25.8	0.78	16.0	0.59	18.7
	215	1.17	21.9	0.60	20.2	0.65	11.2	0.56	11.8

Table 7.11: Power dissipation under 15% inter-die variation and 5% intra-die variation for ISCAS'85 benchmark circuits by various LP models. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively.

delay (Figure 7.10(b), (c), and (d)), clearly "Opt2" is able to suppress the variation of critical delay due to the process-variation for most circuits. Again, this is due to the way we construct our LP model that leads to the simultaneous optimization of both power variation and delay variation. Therefore, we can say that our LP model based on the statistical timing analysis can lead to an optimized circuit, which is resistant to both power and critical delay variations that might be otherwise caused by the process-variation.

7.4 Summary

In this chapter, the experimental results for our process-variation-resistant LP models are given. To illustrate the effectiveness of our methods, results under two different degrees of process-variation are examined with both an example circuit and the ISCAS'85 benchmark circuits. Results show that an optimized circuit obtained from our LP models is able to maintain a low power dissipation under process-variation. Furthermore, our LP model based on statistical timing achieves a solution that is more process-variation-resistant in terms of both power and delay performance.

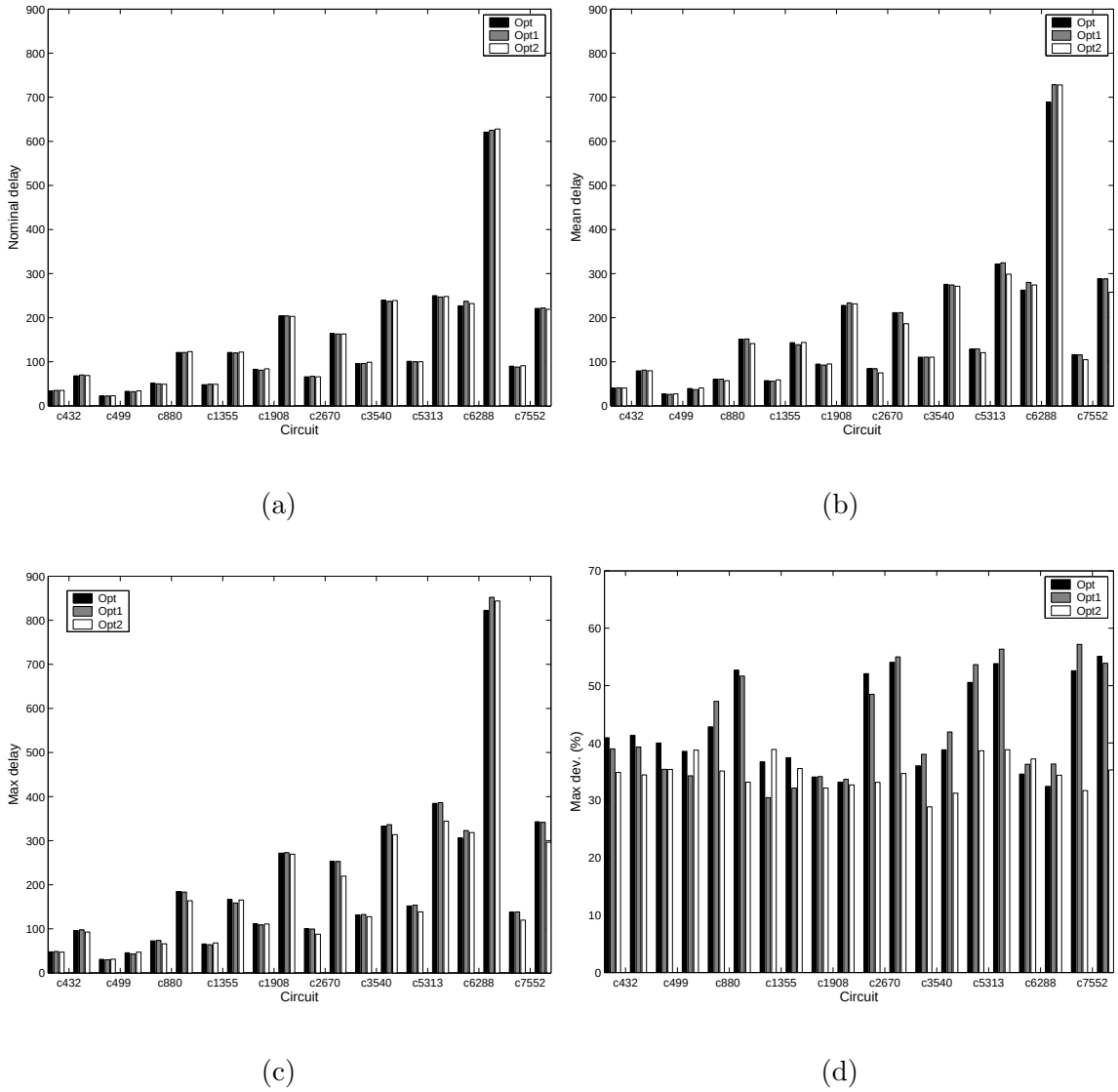


Figure 7.10: Critical delay for optimized ISCAS'85 benchmark circuits under 15% inter-die and 5% intra-die delay variation by various LP models: (a) the nominal critical delays under no process-variation, (b) the mean values of critical delay under the process variation, (c) the maximum values of critical delay under the process variation, (d) the maximum deviation of critical delay (in percentage) from the nominal delay under the process variation. "Opt", "Opt1", and "Opt2" represents the optimization given by LP models in [170], Chapter 4, and Chapter 5, respectively. For each circuit, delay results for two different *maxdelay* parameters are shown.

Chapter 8

Results Analysis for Input-Specific Optimizations

In Chapter 6, we have proposed the input-specific optimization. In this chapter, results from two different input-specific optimization methods are illustrated. The input-specific optimization is first applied to the previous LP model [170] that considers no process-variation. Then we show the results obtained from the application of input-specific optimization to the LP model proposed in Chapter 5. The experimental procedure is same as that in Chapter 7. We show that our input-specific optimization methods are able to achieve the same reduction in power dissipation but require less overhead in terms of number of buffers inserted. We first show results for a smaller example circuit in greater detail and then present summary results for ISCAS'85 benchmark circuit.

8.1 Results for an example circuit

To give a complete analysis of our input-specific optimization, we use the c432 circuit from the ISCAS'85 benchmarks as an example. Two input-specific optimization methods are illustrated. "IS-Opt" is the application of input-specific optimization to the previous LP model [170] that assumes no process-variation. "IS-Opt2" is the application of input-specific optimization to the LP model proposed in Chapter 5. Power estimation is done with 32 stuck-at-fault test vectors (a complete gate level test set).

8.1.1 Input-specific optimization under no process-variation

The power dissipation and critical delay for "Opt" and "IS-Opt" are shown in Table 8.1. In this experiment, "IS-Opt" adopts the selective relaxation described in Section 6.3.1. "Maxdelay" is the maximum delay specification parameter used in both models. "Critical Delay" is the actual critical delay of the optimized circuit.

Maxdelay	Un-opt.	Opt			IS-Opt		
	Avg. Pwr.	Avg. Pwr.	Critical Delay	No. of Buffers	Avg. Pwr.	Critical Delay	No. of Buffers
17	1	0.74	17	95	0.74	18	88
34	1	0.74	34	66	0.74	35	66
51	1	0.74	51	63	0.74	52	45
68	1	0.74	68	58	0.74	69	41

Table 8.1: Experimental results for the input-specific optimization of c432 circuit under no process-variations. "Opt" and "IS-Opt" represents the optimization given by LP models of [170] and Section 6.3.1, respectively.

From Table 8.1, we can see that the application of input-specific optimization can reduce the level of overdesign in the optimized circuit and result in lower overhead in terms of number of buffers inserted. The power dissipation of "IS-Opt" is the same as that of "Opt". Only the critical delay of "IS-Opt" increases slightly and this is mostly due to the quantization errors of gate delays.

8.1.2 Input-specific optimization under process-variation

The experimental results for "Opt2" and "IS-Opt2" are shown in Table 8.2. In these experiments, "IS-Opt2" adopts the selective relaxation with the tuning option turned off ($TF = 0$). In Table 8.2(a), "Nom. Pwr." represents the nominal power dissipation assuming no process-variation. "Mean Pwr." represents the mean of the power distribution with process-variations (15% intra-die and 5% inter-die delay variation). "Max Dev." is the

difference ratio between the maximum value of the power distribution and "Nom. Pwr.". Monte-Carlo simulation method was used where the optimized circuit was simulated using 1,000 randomly sampled sets of gate delays. All power values are normalized according to the power dissipation of the un-optimized circuit under no process-variation. In Table 8.2(b), "Nom. Delay" indicates the critical delay of the circuit under no process-variation, the nominal value of critical delay. We show the *mean* and *std. dev.* of the distribution of critical delay under process-variation. We also show the maximum deviation ("Max. Dev.") of the critical delay from its nominal value.

	Un-opt	**Opt2**				**IS-Opt2**			
$\mathbf{D_{max}}$	**Nom. Pwr.**	**Nom. Pwr.**	**Mean Pwr.**	**Max Dev. (%)**	**No. Buf.**	**Nom. Pwr.**	**Mean Pwr.**	**Max Dev. (%)**	**No. Buf.**
25	1.0	0.74	0.84	25.2	95	0.74	0.84	24.7	88
50	1.0	0.74	0.76	11.1	88	0.74	0.76	9.3	81
74	1.0	0.74	0.76	8.4	89	0.74	0.75	7.5	79
99	1.0	0.74	0.74	3.7	106	0.74	0.74	3.3	76

(a)

	Opt2				**IS-Opt2**			
$\mathbf{D_{max}}$	**Nom. Delay**	**Mean**	**Std. Dev.**	**Max Dev. (%)**	**Nom. Delay**	**Mean**	**Std. Dev.**	**Max Dev. (%)**
25	17	20.6	1.14	41.3	18	21.1	1.20	37.1
50	35	40.6	2.18	34.7	36	40.0	2.18	29.3
74	52	59.2	3.33	33.1	52	58.0	3.26	30.3
99	69	79.4	4.43	34.4	69	77.8	4.65	33.0

(b)

Table 8.2: Experimental results for the input-specific optimization of c432 circuit under process variations (15% intra-die variation and 5% inter-die variation): (a) power dissipation and number of buffers inserted by various LP models, (b) nominal values and distributions of critical delay given by various LP models. "Opt2" and "IS-Opt2" represents the optimization given by LP models in Chapter 5 and Section 6.3.2, respectively. $r = 0.15$ in both "Opt2" and "IS-Opt2".

We see that "IS-Opt2" significantly reduces the number of buffers inserted for the same critical delay specification as compared to "Opt2". When $D_{max} = 99$, number of buffers inserted is reduced from 106 to 76, about 30% reduction. On the other hand, the power dissipation and delay distribution of "IS-Opt2" is equivalent or very similar to that of "Opt2". This example illustrates that the input-specific optimization is able to reduce the overhead partly without sacrificing any performance.

Generalized relaxation: As described in Section 6.3.1, we can adopt the generalized relaxation for making a trade off between power dissipation and number of buffers inserted. To illustrate this, we optimize c432 circuit using "IS-Opt2" with the generalized relaxation for $D_{max} = 99$. In Figure 8.1, we shows the relationships between the parameter τ and resulting power dissipation values. The reduction in the number of buffers is also shown in the figure. As expected, the number of buffers inserted into the circuit is reduced as τ increases. The nominal value and the variation of power dissipation increase when τ increases.

8.2 Results for ISCAS'85 benchmark circuits

ISCAS'85 benchmark circuits are optimized using the input-specific optimization methods. Results for "IS-Opt" and "IS-Opt2" are shown. Same sets of vectors as in Chapter 7 were used for power estimation. For completeness, results for c432 circuit are repeated.

8.2.1 Input-specific optimization under no process-variation

The power dissipation and critical delay for "Opt" and "IS-Opt" are shown in Table 8.3. In these experiments, "IS-Opt" adopts the selective relaxation described in Section 6.3.1.

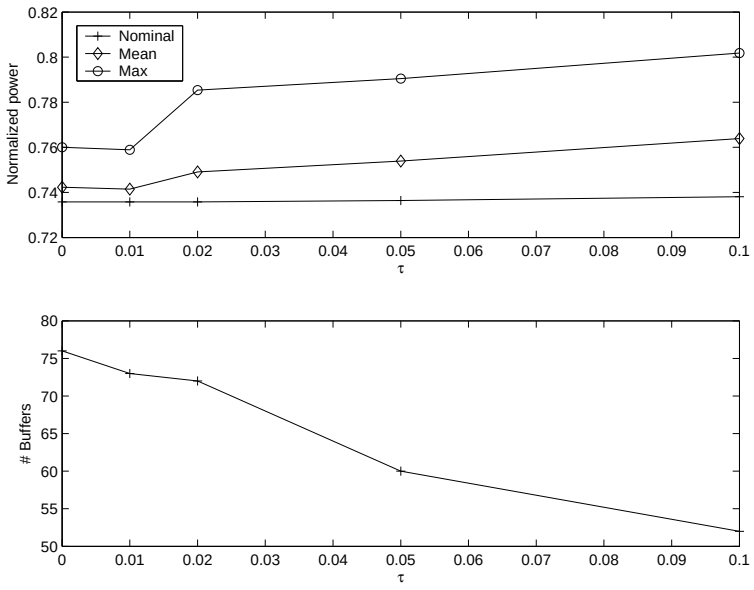


Figure 8.1: Trade-off between power dissipation and number of buffers inserted. c432 circuit is optimized by "IS-Opt2" with the generalized relaxations under $D_{max} = 99$, $r = 0.15$ and varying τ values. In the upper figure, nominal power under no process variation, mean and maximum value of power distribution under the process-variation (15% intra-die variation and 5% inter-die variation) are shown. All power values are normalized according to the power dissipation of the un-optimized circuit under no process-variation. In the lower figure, number of buffers required for the optimization is shown.

Circuit	Max-delay	Un-opt	Opt			IS-Opt		
		Avg. Pwr.	Avg. Pwr.	Critical Delay	No. of Buffers	Avg. Pwr.	Critical Delay	No. of Buffers
c432	34	1.0	0.74	34	66	0.74	35	66
	68	1.0	0.74	68	58	0.74	69	41
c499	22	1.0	0.94	22	48	0.94	22	33
	33	1.0	0.94	33	0	0.95	33	0
c880	48	1.0	0.54	51	35	0.54	49	32
	120	1.0	0.54	121	30	0.54	122	24
c1355	48	1.0	0.93	48	192	0.93	48	113
	120	1.0	0.93	121	128	0.93	120	25
c1908	80	1.0	0.53	82	62	0.54	86	52
	200	1.0	0.54	203	34	0.53	204	3
c2670	64	1.0	0.74	65	34	0.74	66	30
	160	1.0	0.74	163	9	0.74	162	1
c3540	94	1.0	0.59	95	139	0.59	101	122
	235	1.0	0.59	239	78	0.59	239	73
c5315	98	1.0	0.56	100	167	0.56	104	170
	245	1.0	0.56	249	53	0.56	250	52
c6288	228	1.0	0.13	226	870	0.13	228	870
	620	1.0	0.13	620	857	0.13	620	853
c7552	86	1.0	0.52	89	91	0.52	88	84
	215	1.0	0.52	220	44	0.52	221	38

Table 8.3: Experimental results for input-specific optimization of ISCAS'85 benchmark circuits under no process-variations. "Opt" and "IS-Opt" represents the optimization given by LP models of [170] and Section 6.3.1, respectively.

We see that the input-specific optimization is able to reduce the number of buffers inserted while maintaining the same performance in terms of power dissipation and critical delay. The power dissipation and critical delay values are equivalent or very similar for "Opt" and "IS-Opt" in most cases. Depending on the vectors and circuits, a varying degree of improvement is achieved. In a good case, e.g., c1355, the number of buffers is reduced by up to 80%.

8.2.2 Input-specific optimization under process-variation

Power analysis

Power dissipation and number of buffers inserted by "Opt2" and "IS-Opt2" are shown in Table 8.4. In these experiments, "IS-Opt2" adopts the selective relaxation. The tuning option is turned on only for c1908, c3540, and c6288, where TF is chosen to be $\frac{1}{D_{max}}$. In these experiments, 15% intra-die and 5% inter-die delay variation were assumed ($r = 0.15$ for "Opt2" and "IS-Opt2").

		Un-opt	Opt2				IS-Opt2			
Cir.	**D_{max}**	**Nom. Pwr.**	**Nom. Pwr.**	**Mean Pwr.**	**Max Dev. (%)**	**No. Buf.**	**Nom. Pwr.**	**Mean Pwr.**	**Max Dev. (%)**	**No. Buf.**
c432	50	1.0	0.74	0.76	11.1	88	0.74	0.76	9.3	81
	99	1.0	0.74	0.74	3.7	106	0.74	0.74	3.3	76
c499	32	1.0	0.94	0.95	2.0	88	0.94	0.95	1.9	88
	48	1.0	0.94	0.95	1.0	129	0.94	0.95	1.8	58
c880	70	1.0	0.54	0.59	18.2	57	0.54	0.59	20.4	38
	174	1.0	0.54	0.55	8.6	62	0.54	0.56	9.0	38
c1355	70	1.0	0.93	0.98	10.2	305	0.93	1.01	13.1	253
	174	1.0	0.93	0.94	3.0	305	0.93	0.95	4.7	160
c1908	116	1.0	0.52	0.64	35.8	135	0.52	0.64	34.7	107
	290	1.0	0.52	0.58	21.4	190	0.52	0.57	18.4	104
c2670	93	1.0	0.74	0.80	13.6	249	0.73	0.79	11.3	186
	232	1.0	0.73	0.76	6.2	211	0.73	0.75	4.3	79
c3540	137	1.0	0.59	0.66	17.8	281	0.59	0.65	15.6	247
	341	1.0	0.59	0.62	10.1	311	0.59	0.61	7.4	188
c5315	143	1.0	0.55	0.63	20.8	399	0.55	0.63	21.0	389
	356	1.0	0.55	0.60	13.4	418	0.55	0.60	13.2	413
c6288	331	1.0	0.13	0.38	223.8	1121	0.13	0.38	225.2	1115
	899	1.0	0.13	0.26	125.3	1473	0.13	0.26	125.5	1243
c7552	125	1.0	0.52	0.59	18.7	481	0.52	0.58	18.1	389
	312	1.0	0.52	0.56	11.8	645	0.52	0.55	10.9	520

Table 8.4: Power dissipations and number of buffers inserted by the input-specific optimizations of ISCAS'85 benchmark circuits under process variations (15% intra-die variation and 5% inter-die variation). "Opt2" and "IS-Opt2" represents the optimization given by LP models in Chapter 5 and Section 6.3.2, respectively. $r = 0.15$ in both "Opt2" and "IS-Opt2".

We see that in all cases power dissipation of optimized circuits by "Opt2" and "IS-Opt2" is either equivalent or has only a slight difference. However, "IS-Opt2" is able to achieve a solution with a smaller number of buffers inserted. The reduction of buffers is more obvious for larger D_{max} for each circuit. This is because, for a smaller D_{max}, path balancing is more difficult. Removing of glitch-filtering constraint has a smaller effect on the reduction of buffers. In these experiments, up to 63% reduction in the number of buffers is achieved for c2670 circuit.

Delay analysis

The critical delays under process-variation are shown in Figure 8.2. We see that "Opt2" and "IS-Opt2" have equivalent performances in all cases. From the power dissipation results in Table 8.4 and this figure, we can conclude that our input-specific optimization method "IS-Opt2" achieves a better solution for a given input sequence. It is able to maintain the same power and delay performance while reducing the overhead in terms of the number of buffers inserted.

8.3 Summary

In this chapter, experimental results for our input-specific optimization methods are given. Our input-specific optimization is applied to a previous LP model [170] and our process-variation-resistant LP model of Chapter 5. Experimental results show that the input-specific optimization methods obtain better solutions with lower overhead in terms of the number of buffers inserted while maintaining the same power delay performance.

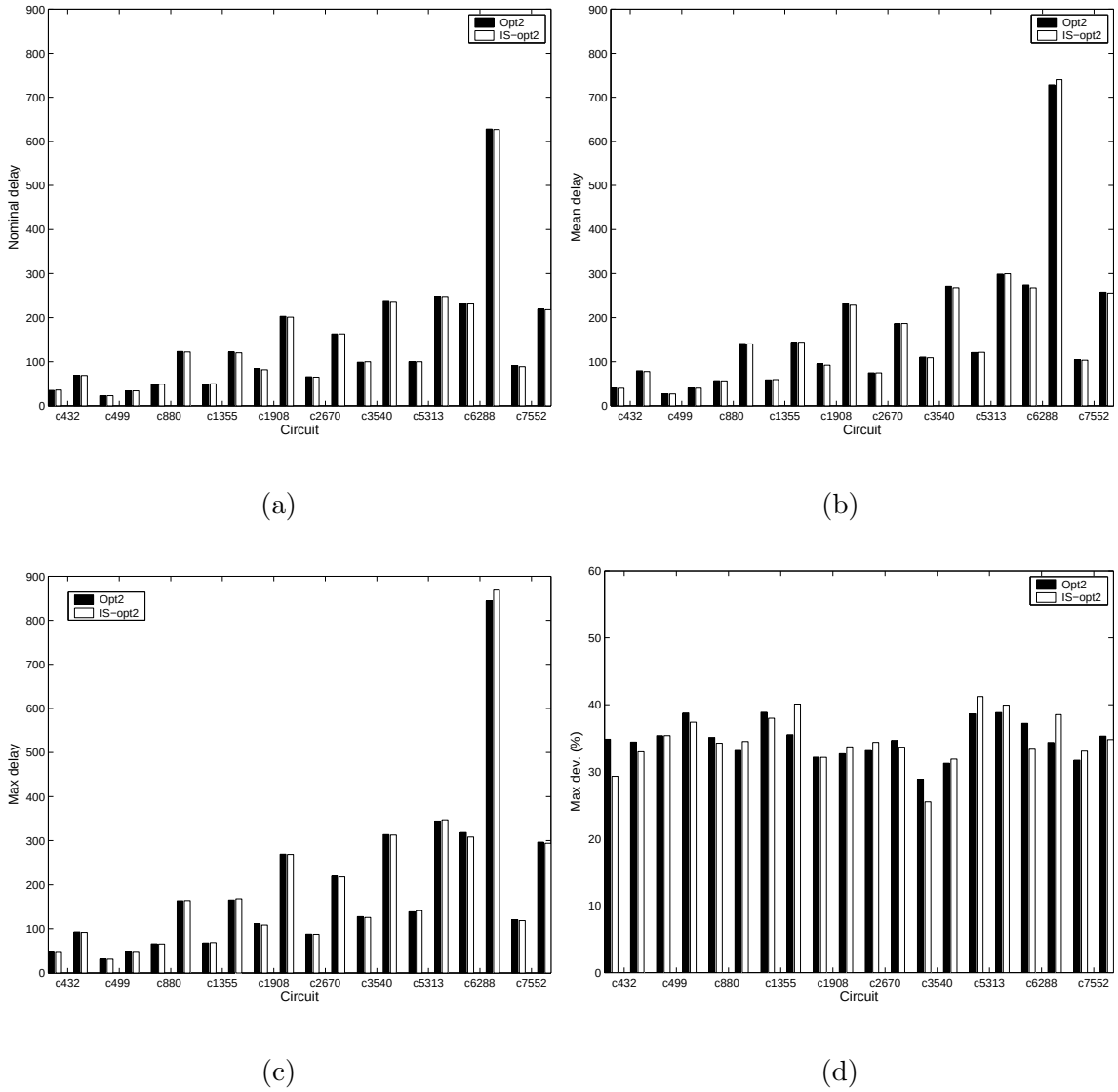


Figure 8.2: Critical delays for ISCAS'85 benchmark circuits under 15% inter-die and 5% intra-die delay variation by the input-specific optimization: (a) the nominal critical delays under no process-variation, (b) the mean values of critical delay under the process variation, (c) the maximum values of critical delay under the process variation, (d) the maximum deviation of critical delay (in percentage) from the nominal delay under the process variation. "Opt2", "IS-Opt2" represents the optimization given by LP models in Chapter 5 and Section 6.3.2, respectively. For each circuit, delay results for two different D_{max} parameters are shown.

Chapter 9

Conclusion and Future Work

In this chapter, we conclude our work and discuss possible future work.

9.1 Conclusion

Low power dissipation has become a crucial factor in the modern VLSI circuit design. Linear programming techniques [4, 170, 171] have been proposed to reduce the switching power of a circuit by removing glitches. However, the prior techniques have limitations due to the fixed-delay assumption. In this dissertation, we provide new optimization methods considering the effects of process variation.

As variability of process and circuit parameters keep increasing when technology scales into the deep-sub-micron regime, gate delay will not be a constant as assumed before. The variations of gate delays can easily corrupt the optimality of the solution given by previous techniques. In this dissertation, we propose process-variation-resistant LP models. Gate delays are not deterministic values but are modeled as random variables. The effect of process-variation is considered in terms of delay variations. Two basic types of process-variation are considered in our analysis: inter-die variations and intra-die variations. We prove that the effect of inter-die variations on the switching power dissipation is negligible and construct two LP models based on the worst-case timing analysis and the statistical timing analysis individually.

Experimental results have shown that our process-variation-resistant models can lead to solutions that are robust under process-variations. Both power dissipation and critical delay

distribution have a smaller deviation from their nominal values under process-variations. Experimental results have shown that our LP model based on statistical timing performs better than the LP model based on the worst-case timing, especially under larger process-variation. Under a 15% intra-die variation and 5% inter-die variation a given critical delay specification, the power dissipation of the optimized circuit by a previous method [170] can be up to 2.64 times of the design value. Our process-variation-resistant optimization is able to reduce this deviation by 53%. For certain smaller circuits, e.g., c499, deviation of power dissipation can be almost completely suppressed without increasing the circuit delay specification. In most cases, our LP model based on statistical timing can reduce the deviation of critical delay in the optimized circuit. In our experiments, up to 40% reduction of "Max. Dev." is achieved.

To reduce the number of delay elements inserted into the optimized circuit, we consider optimizing the circuit for a given input sequence that may be specified for the circuit. In the input-specific optimization, we relax the constraints for gates where glitches are unlikely to occur. We define the concept of glitch-generation pattern and glitch-generation probability. By observing the glitch-generation probability for each gate, we can adaptively relax the glitch-filtering constraint. The experimental results show that we are able to obtain a better solution with fewer buffer insertions while maintaining the similar power reduction as before. In our experiments, the application of input-specific optimization to the LP model of Chapter 5 is able to reduce the number of buffers by up to 63%.

9.2 Future work

In this section, some thoughts on future work are given. These ideas may serve as proposals on the possible work that could be done later.

9.2.1 Gate sizing

The technique described in this dissertation is restricted to the derivation of gate delays that can lead to a more robust circuit under process-variation. The actual device sizes for each gate could be obtained via the technical mapping method proposed in [169]. In that approach gate sizes are determined from primary outputs to primary inputs using a *reverse breath-first* search algorithm. Therefore, the reduction of dynamic power from our approach is mostly obtained by reduction of glitches.

However, to reduce the total dynamic power, glitch reduction alone may not be sufficient. The total load capacitance should also be reduced. This can be done through gate sizing. Gate sizing technique is generally not preferred because it normally suffers from the non-linearity problems of the delay model and cannot be solved using a linear program. However, there is evidence showing the possibility of using linear program in gate sizing for power optimization. Berkerlaar et al. [24] proposed a gate sizing method using a linear gate delay model. Piece-wise linear approximation is adopted to convert a non-linear gate delay model to a linear form. Mani et al. [128] presented a statistical sizing approach that takes into account randomness in gate delays by formulating an efficient linear program. The same linear gate delay model is used. In all these approaches, the reduction of glitches is not considered. Power reduction is obtained by minimizing the total area.

It might be possible to devise a linear program using the linear gate delay model proposed by Berkerlaar [24]. This linear program will be able to minimize the dynamic power dissipation considering both total capacitance and glitch reduction. Process-variation-resistance will be one more feature that can be added.

9.2.2 Routing delay

As technology scales, VLSI circuits are getting more and more interconnects dominant. The routing delay and routing capacitance play a more important role in a VLSI chip. In our LP approach, routing delay is lumped together with gate inertial delay and is represented with a single variable d_i. During the later transistor/gate sizing step, an iterative approach may be necessary to map the delay assignments to physical dimensions of gates considering routing delays.

In addition, routing delay may impose some lower bound limit on d_i and delay assignments may have to be re-generated after the layout is done. Such iterative approach can be considered if we want to construct a complete scheme that combines LP approach (for delay assignments) and transistor sizing (for realization of delay assignment) to generate the final physical level design.

9.2.3 Delay element

In our optimization, the delay elements or buffers inserted in the circuit are assumed to be of resistive type and do not consume additional power. In reality, even the resistive feedthrough cell proposed in [204, 205] consumes some additional power. Investigations on delay element that produce same amount of delay with less power consumption and area overhead will be useful. It is also possible to extend our work using the variable-input-delay gate proposed in [169] to avoid the insertion of buffers.

9.2.4 Leakage power

Our process-variation-resistant optimization is helpful in the reduction of leakage power variation. The two major sources of leakage power variation are the variation of threshold

voltage V_t and thermal voltage V_T. The thermal voltage varies when temperature changes. The leakage current variation has an exponential relationship with these two parameters. While temperature of a gate is mostly determined by its power dissipation, the reduction of the variation in dynamic power has direct impact on the variation of the operating temperature. Thus, leakage power variation is suppressed when dynamic power variation is suppressed. Further research incorporating our technique into the reduction of leakage power variation might be possible.

Bibliography

[1] I. S. Abu-Khater, A. Bellaouar, and M. I. Elmasry, "Circuit techniques for CMOS low-power high-performance multipliers," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 10, pp. 1535–1546, 1996.

[2] A. Agarwal, V. Zolotov, and D. T. Blaauw, "Statistical timing analysis using bounds and selective enumeration," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 9, pp. 1243–1259, 2003.

[3] V. D. Agrawal, "Low power disign by hazard filtering," in *Proceedings of the International Conference on VLSI Design*, Jan. 1997, pp. 193–197.

[4] V. D. Agrawal, M. L. Bushnell, G. Parthasarathy, and R. Ramadoss, "Digital circuit design for mimimum transient energy and linear programming method," in *Proceedings of the International Conference on VLSI Design*, Jan. 1999, pp. 434–439.

[5] M. W. Allam, M. H. Anis, and M. I. Elmasry, "High-speed dynamic logic styles for scaled-down CMOS and MTCMOS technologies," in *Proceedings of the 2000 International Symposium on Low Power Electronics and Design, ISLPED00*, 2000, pp. 155–160.

[6] M. J. Ammer, M. Bolotski, P. Alvelda, and T. F. Knight, "A 160x120 pixel liquid-crystal-on-silicon microdisplay with an adiabatic DACM," in *IEEE Solid-State Circuits Conference*, Nov. 1999, pp. 212–213.

[7] 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.

[8] W. Athas and L. J. Svensson, "Reversible logic issues in adiabatic computing," in *IEEE workshop on Physics and Computation, PhysComp'94*, Nov. 1994, pp. 111–118.

[9] W. Athas, L. J. Svensson, J. G. Koller, N. Tzartzanis, and E. Chou, "Low-power digital systems based on adiabatic-switching principles," *IEEE Transaction on VLSI Systems*, pp. 398–407, Dec. 1994.

[10] W. C. Athas, L. J. Svensson, and N. Tzartzanis, "A resonant signal driver for two-phase, almost-nonoverlapping clocks," in *Proceedings of IEEE International Symposium on Circuits and Systems, ISCAS'96*, 1996, pp. 129–132.

[11] W. C. Athas, N. Tzartzanis, W. Mao, L. Peterson, R. Lal, K. Chong, J.-S. Moon, L. Svensson, and M. Bolotski, "The design and implementation of a low-power clock-powered microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 35, no. 11, pp. 1561–1570, Nov. 2000.

[12] W. C. Athas, N. Tzartzanis, L. J. Svensson, and L. Peterson, "A low-power microporcessor based on resonant energy," *IEEE Journal of Solid-State Circuits*, vol. SC-32, no. 11, pp. 1693–1701, Nov. 1997.

[13] X. Bai, C. Visweswariah, P. Strenski, and D. Hathaway, "Uncertainty aware circuit optimization," in *Proceedings of ACM/IEEE Design Automation Conference*, 2002, pp. 58–63.

[14] A. Bellaouar and M. I. Elmasry, *Low-power Digital VLSI Design: Circuits and Systems*. Boston: Kluwer Academic Publisher, 1995.

[15] L. Benini, "Leading edge low power design [SoCs]," in *Proceedings of Design Automation Conference, Asia and South Pacific (ASP-DAC 2003)*, Jan. 2003, pp. 385–389.

[16] L. Benini, A. Bogliolo, and G. DeMicheli, "Regression models for behavioral power estimation," in *Proceedings of the International Workshop on Power and Timing Modeling, Optimization and Simulation*, Sept. 1996, pp. 179–186.

[17] L. Benini, A. Bogliolo, and G. D. Micheli, "A survey of design techniques for system-level dynamic power management," *IEEE Transactions on Very Large-scale Integration Systems*, vol. 8, no. 3, pp. 299–316, 2000.

[18] C. Bennett, "Logic reversibility of computation," *IBM Journal of Research & Development*, vol. 17, pp. 525–532, 1973.

[19] C. Bennett, "Time/space trade-offs for reversible computation," *SIAM Journal of Computing*, vol. 18, pp. 766–776, 1989.

[20] C. Bennett and R. Landauer, "The fundamental physical limits of computation," *Scientific American*, pp. 48–56, July 1985.

[21] M. Berkelaar, "Statistical delay calculation, a linear time method," in *Proceedings of TAU 97*, Dec. 1997, pp. 15–24.

[22] M. Berkelaar, P. Buurman, and J. Jess, "Computing entire area/power consumption versus delay trade-off curve for gate sizing using a piecewise linear simulator," *IEEE Transactions on Computer Aided Design of Circuits and Systems*, vol. 15, no. 11, pp. 1424–1434, Nov. 1996.

[23] M. Berkelaar and E. Jacobs, "Using gate sizing to reduce glitch power," in *Proceedings of the ProRISC Workshop on Circuits, Systems and Signal Processing, (Mierlo, The Netherlands)*, Nov. 1996, pp. 183–188.

[24] M. Berkelaar and J. A. G. Jess, "Gate sizing in MOS digital circuits with linear programming," in *Proceedings of the European Design Automation Conference, Mierlo, The Netherlands*, Mar. 1990, pp. 217–221.

[25] M. Borah, M. J. Irwin, and R. M. Owens, "Minimizing power consumption of static CMOS circuits by transistor sizing and input reordering," in *Proceedings of the International Conference on VLSI Design*, Jan. 1995, pp. 294–298.

[26] M. Borah, R. Owens, and M. Irwin, "Transistor sizing for low power CMOS circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 6, pp. 665 – 671, 1996.

[27] R. Brawhear, N. Menezes, C. Oh, L. Pillage, and R. Mercer, "Predicting circuit performance using circuit-level statistical timing analysis," in *Proceedings of European Design and Test Conference*, 1994, pp. 332–337.

[28] R. Bryant, "A switch-level model and simulator for MOS digital systems," *IEEE Transactions on Computers*, vol. 33, no. 2, pp. 160–177, 1984.

[29] R. E. Bryant, "Graph-based algorithm for boolean function manipulation," *IEEE Transaction on Computers*, vol. 35, no. 8, pp. 677–691, 1986.

[30] R. Burch, F. Najm, P. Yang, and D. Hocevar, "Pattern-independent current estimation for reliability analysis of CMOS circuits," in *Proceedings of 25th ACM/IEEE Design Automation Conference, Anaheim, CA*, June 1988, pp. 294–299.

[31] R. Burch, F. N. Najm, P. Yang, and T. N. Trick, "A Monte Carlo approach for power estimation," *IEEE Transaction on VLSI Systems*, vol. 1, no. 1, pp. 63–71, Mar. 1993.

[32] A. Chandrakasan, M. Potkonjak, J. Rabaey, and R. W. Brodersen, "HYPER-LP: A system for power minimization using architectural transformation," in *Proceedings of the IEEE International Conference on Computer Aided Design*, 1992, pp. 300–303.

[33] A. P. Chandrakasan and R. W. Brodersen, *Low power digital CMOS design*. Boston: Kluwer academic publishers, 1995.

[34] A. P. Chandrakasan and R. W. Brodersen, *Low power digital CMOS design*, chapter 7, pp. 249–254. Boston: Kluwer academic publishers, 1995.

[35] A. P. Chandrakasan and R. W. Brodersen, *Low power digital CMOS design*, chapter 6, pp. 181–218. Boston: Kluwer academic publishers, 1995.

[36] H. Y. Chen and S. M. Kang, "ICOACH: A circuit optimization aid for CMOS high-performance circuits," *Integration, the VLSI Journal*, vol. 10, no. 2, pp. 185–212, 1991.

[37] W.-K. Chen, editor, *The VLSI handbook*, chapter 18, pp. 18–6 – 18–10. CRC Press, 2000.

[38] Z. Chen, C. Diaz, J. Plummer, M. Cao, and W. Greene, "0.18 μm dual Vt MOSFET process and energy-delay measurement," in *1996 IEEE International Electron Devices Meeting, Technical Digest*, Dec. 1996, pp. 851–854.

[39] K. T. Cheng and V. D. Agrawal, "An entropy measure for the complexity of multi-output boolean functions," in *Proceedings of ACM/IEEE Design Automation Conference, Orlando, FL*, June 1990, pp. 302–305.

[40] G. R. Cho and T. Chen, "On mixed PTL/static logic for low-power and high-speed circuits," *VLSI Design : An International Journal of Custom-Chip Design, Simulation, and Testing*, vol. 12, no. 3, pp. 399–406, 2001.

[41] G. R. Cho and T. Chen, "Mixed PTL/static logic synthesis using genetic algorithms for low-power applications," in *Proceedings of International Symposium on Quality Electronic Design*, March 2002, pp. 458–463.

[42] G. R. Cho and T. Chen, "On the impact of technology scaling on mixed PTL/static circuits," in *Proceedings of IEEE International Conference on Computer Design: VLSI in Computers and Processors*, Sept 2002, pp. 322–326.

[43] K. Chu and D. Pulfrey, "A comparison of CMOS circuit techniques: Differential cascode voltage switch logic versus conventional logic," *IEEE Journal of Solid-State Circuits*, vol. 22, no. 4, pp. 528–532, 1987.

[44] M. A. Cirit, "Estimating dynamic power consumption of CMOS circuits," in *Proceedings of IEEE International Conference on Computer-Aided Design*, Nov. 1987, pp. 534–537.

[45] B. Coates, A. Davis, and K. Stevens, "The post office experience: Designing a large asynchronous chip," *Integration, the VLSI Journal*, vol. 15, no. 3, pp. 341–366, 1993.

[46] J. Compton and A. Albicki, "Self timed pipeline with adder," in *Proceedings of the 2nd Great Lakes Symposium on VLSI*, 1992, pp. 109–113.

[47] A. R. Conn, P. K. Coulman, R. A. Haring, G. L. Morrill, C. Visweshwariah, and C. W. Wu, "JiffyTune: Circuit optimization using time-domain sensitivities," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, pp. 1292–1309, Dec. 1998.

[48] P. Corsonello, S. Perri, and G. Cocorullo, "Performance comparison between static and dynamic CMOS logic implementations of a pipelined square-rooting circuit," *IEE Proceedings of Circuits, Devices and Systems*, vol. 147, no. 6, pp. 347–355, 2000.

[49] Z. Dai and K. Asada, "MOSIZ: A two-step transistor sizing algorithm based on optimal timing assignment method for multi-stage complex gates," in *Proceedings of 1989 Custom Integrated Circuits Conference*, May 1989, pp. 17.3.1–17.3.4.

[50] G. B. Dantzig, "Programming of interdependent activities. II. mathematical model," *Econometrica*, vol. 17, pp. 200–211, 1949.

[51] I. David, R. Ginosar, and M. Yoeli, "An efficient implementation of Boolean functions as self timed circuits," *IEEE Transaction on Computers*, vol. 41, no. 1, pp. 2–10, 1992.

[52] V. De and S. Borkar, "Technology and design challenges for low power and high performance," in *Proceedings of 1999 IEEE Symposium on Low Power Electronics and Design, San Diego, CA*, Aug. 1999, pp. 163–168.

[53] Y. Deguchi, N. Ishiura, and S. Yajima, "Probabilistic CTSS: Analysis of timing error probability in asynchronous logic circuits," in *Proceedings IEEE/ACM Design Automation Conference*, 1991, pp. 650–655.

[54] C. Deng, "Power analysis for CMOS/BiCMOS circuits," in *Proceedings of the 1994 International Workshop on Low Power Design*, Nov. 1994, pp. 3–8.

[55] J. S. Denker, "A review of adiabatic computing," in *Proceedings of IEEE Symposium on Low Power Electronics, San Diego, CA*, 1994, pp. 94–95.

[56] S. Devadas, H. F. Jyu, K. Keutzer, and S. Malik, "Statistical timing analysis of combinational circuits," in *Proceedings of IEEE International Conference on Computer Design*, 1992, pp. 38– 43.

[57] A. G. Dickinson and J. S. Denker, "Adiabatic dynamic logic," *IEEE Journal of Solid-State Circuits*, vol. 30, pp. 311–315, 1995.

[58] C.-S. Ding, C.-Y. Tsui, and M. Pedram, "Gate-level power estimation using tagged probabilistic simulation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 11, pp. 1099–1107, Nov. 1998.

[59] D. Duarte, Y.-F. Tsai, N. Vijaykrishnan, and M. J. Irwin, "Evaluating run-time techniques for leakage power reduction," in *Proceedings of ACM/IEEE Design Automation Conference, Asia South Pacific, 2002*, 2002, pp. 31–38.

[60] S. Dutta, S. Nag, and K. Roy, "ASAP: A transistor sizing tool for area, delay and power optimization of CMOS circuits," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, May 1994, pp. 61–64.

[61] M. Elgebaly and M. Sachdev, "Efficient adaptive voltage scaling system through on-chip critical path emulation," in *Proceedings of the 2004 International Symposium on Low Power Electronics and Design, ISLPED '04*, Aug. 2004, pp. 375 – 380.

[62] M. E. S. Elraba, M. H. Anis, and M. I. Elmasry, "A contention-free domino logic for scaled-down CMOS technologies with ultra low threshold voltages," in *Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS 2000*, volume 1, May 2000, pp. 748–751.

[63] S. Ercolani, M. Favalli, M. Damiani, P. Olivo, and B. Ricco, "Estimate of signal probability in combinational logic networks," in *Proceedings of the First European Test Conference*, 1989, pp. 132–138.

[64] C. Farnsworth, D. Edwards, J. Liu, and S. Sikand, "A hybrid asynchronous system design environment," in *Proceedings of the 2nd working conference on Asynchronous Design Methodologies*, May 1995, pp. 91–98.

[65] F. Ferrandi, F. Fummi, E. Macii, M. Poncino, and D. Sciuto, "Power estimation of behavioral descriptions," in *Proceedings of IEEE Design Automation and Test in Europe, Paris, France*, Feb. 1998, pp. 762–766.

[66] J. Fishburn and S. Taneja, "Transistor sizing for high performance and low power," in *Proceedings of the IEEE 1997 Custom Integrated Circuits Conference*, May 1997, pp. 591–594.

[67] J. P. Fishburn and A. E. Dunlop, "TILOS: A posynomial programming approach to transistor sizing," in *Procceedings of IEEE International Conference on Computer-Aided Design*, Nov. 1985, pp. 326–328.

[68] R. Fourer, D. M. Gay, and B. M. Kernighan, *AMPL: A modeling languague for mathematical programming.* South San Francisco, California: The scientific press, 1993.

[69] S. Furber, "Computing without clocks: Micropipelining the ARM processor," in *Asynchronous Digital Circuit Design (Workshops in Computing)*, (New York), Springer-Verlag, 1995, pp. 211–262.

[70] S. B. Furber, J. D. Garside, P. Riocreux, S. Temple, P. Day, J. Liu, and N. C. Paver, "AMULET2e: An asynchronous embedded controller," *Proceedings of the IEEE*, vol. 87, no. 2, pp. 243–256, 1999.

[71] T. Gabara, "Pulsed low power CMOS," *International Journal of High Speed Electronics and Systems*, vol. 5, no. 2, pp. 177–182, 1994.

[72] J. Garside, W. Bainbridge, A. Bardsley, D. Clark, D. Edwards, S. Furber, J. Liu, D. Lloyd, S. Mohammadi, J. Pepper, O. Petlin, S. Temple, and J. Woods, "AMULET3i-an asynchronous system-on-chip," in *Proceedings of the Sixth International Symposium on Advanced Research in Asynchronous Circuits and Systems, (ASYNC 2000)*, April 2000, pp. 162–175.

[73] J. D. Garside, S. Temple, and R. Mehra, "The AMULET2e cache systems," in *Proceedings of International Symposium on Advanced Research in Asynchronous Circuits and Systems*, Mar. 1996, pp. 208–217.

[74] A. Gattiker, S. Nassif, R. Dinakar, and C. Long, "Timing yield estimation from static timing analysis," in *Proceedings of the International Symposium on Quality Electronic Design*, 2001, pp. 437–442.

[75] A. Ghosh, S. Devadas, K. Keutzer, and J. White, "Estimation of average switching activity in combinational and sequential circuits," in *Proceedings of the 29th ACM/IEEE Design Automation Conference, Anaheim, CA*, June 1992, pp. 253–259.

[76] D. Green, *Modern Logic Design*, pp. 15–17. Addison-Wesley, 1986.

[77] P. Gronowski, W. Bowhill, R. Preston, M. Gowan, and R. Allmon, "High-performance microprocessor design," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 5, pp. 678–686, 1998.

[78] S. Gupta and F. N. Najm, "Power macromodeling for high-level power estimation," in *Proceedings of ACM/IEEE Design Automation Conference, Anaheim, CA*, June 1997, pp. 365–370.

[79] M. Hashimoto and H. Onodera, "Post-layout transistor sizing for power reduction in cell-based design," in *Proceedings of Design Automation Conference, Asia and South Pacific (ASP-DAC 2001)*, Jan. 2001, pp. 359–365.

[80] M. Hashimoto, H. Onodera, and K. Tamaru, "A practical gate resizing technique considering glitch reduction for low power design," in *Proceedings of the 36th Design Automation Conference*, June 1999, pp. 446 – 451.

[81] J. Hayes, "An introduction to switch-level modeling," *IEEE Design and Test of Computers*, vol. 4, no. 4, pp. 18–25, 1987.

[82] R. B. Hitchcock, "Timing verification and the timing analysis program," in *Proceedings of IEEE/ACM Design Automation Conference*, 1982, pp. 594–604.

[83] C.-T. Hsieh, C.-S. Ding, Q. Wu, and M. Pedram, "Statistical sampling and regression estimation in power macro-modeling," in *Proceedings of IEEE/ACM International Conference on Computer Aided Design, ICCAD-96, San Jose, CA*, Nov. 1996, pp. 583–588.

[84] C.-T. Hsieh, M. Pedram, H. Mehta, and F. Rastgar, "Profile-driven program synthesis for evaluation of system power dissipation," in *Proceedings of ACM/IEEE Design Automation Conference, Anaheim, CA*, June 1997, pp. 576–581.

[85] F. Hu and V. D. Agrawal, "Dual-transition glitch filtering in probabilistic waveform power estimation," in *Proceedings of the 15th ACM Greate Lakes Symposium on VLSI*, April 2005, pp. 357–360.

[86] F. Hu and V. D. Agrawal, "Enhanced dual-transition probabilistic power estimation with selective supergate analysis," in *Proceedings of 23rd International Conference on Computer Design (ICCD 2005)*, Oct. 2005, pp. 366–369.

[87] B. Hunt, K. Stevens, B. Suter, and D. Gelosh, "A single chip low power asynchronous implementation of an FFT algorithm for space applications," in *Proceedings of the Fourth International Symposium on Advanced Research in Asynchronous Circuits and Systems*, 30 Mar. – 2 April 1998, pp. 216–223.

[88] ITRS, "2004 international technology roadmap for semiconductors." Semiconductor Industrial Association. http://www.itrs.net/Common/2004Update/2004Update.htm.

[89] E. Jacobs and M. Berkelaar, "Gate sizing using a statistical delay model," in *Proceedings of Design, Automation and Test in Europe Conference and Exhibition*, March 2000, pp. 283–290.

[90] G. Jacobs and R. W. Brodersen, "A fully asynchronous digital signal processor using self-timed circuits," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 6, pp. 1526–1537, 1990.

[91] Y. Jiang, S. S. Sapatneker, and C. Bamji, "Technology mapping for high performance static CMOS and pass transistor logic designs," Technical report, Dept. of ECE, Iowa State University, 1999.

[92] M. C. Johnson, D. Somasekhar, and K. Roy, "Leakage control with efficient use of transistor stacks in single threshold CMOS," in *Proceedings of the ACM/IEEE Design Automation Conference*, 1999, pp. 442–445.

[93] N. P. Jouppi, "Timing analysis for nMOS VLSI," in *Proceedings of the ACM/IEEE Design Automation Conference*, 1983, pp. 411–418.

[94] H. F. Jyu and S. Mahk, "Statistical timing optimization of combinational logic circuits," in *Proceedings of IEEE International Conference on Computer Design*, 1993, pp. 77–80.

[95] A. Kahng and Y. Pati, "Subwavelength optical lithography: Challenges and impacts on physical design," in *Proceedings of ACM Internation Symposium on Physical Design*, 1999, pp. 112–119.

[96] S. M. Kang, "Accurate simulation of power dissipation in VLSI circuits," *IEEE Journal of Solid-State Circuits*, vol. 21, no. 5, pp. 889–891, 1986.

[97] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, vol. 4, pp. 373–395, 1984.

[98] A. Keshavarzi, S. Ma, S. Narendra, B. Bloechel, K. Mistry, T. Ghani, S. Borkar, and V. De, "Effectiveness of reverse body bias for leakage control in scaled dual Vt CMOS ICs," in *Proceedings of International Symposium on Low Power Electronics and Design*, Aug. 2001, pp. 207–212.

[99] J. Kessels and P. Marston, "Designing asynchronous standby circuits for a low-power pager," *Proceedings of the IEEE*, vol. 87, no. 2, pp. 257–267, Feb. 1999.

[100] J. Kessels and A. Peeters, "The Tangram framework: Asynchronous circuits for low power," in *Proceedings of Design Automation Conference, Asia and South Pacific*, 2001, pp. 255–260.

[101] L. G. Khachian, "A polynomial algorithm in linear programming," *Soviet Math. Dokl.*, vol. 20, pp. 191–194, 1979.

[102] J. Kim, C. H. Ziesler, and M. C. Papaefthymiou, "Energy recovering static memory," in *Proceedings of IEEE Symposium on Low Power Electronics and Design*, Aug. 2002, pp. 92–97.

[103] K. Kim, P. Beerel, and Y. Hong, "An asynchronous matrix-vector multiplier for discrete cosine transform," in *Proceedings of the 2000 International Symposium on Low Power Electronics and Design, ISLPED'00*, July 2000, pp. 256–261.

[104] S. Kim, J. Kim, and S.-Y. Hwang, "New path balancing algorithm for glitch power reduction," *IEE Proceedings of Circuits, Devices and Systems*, vol. 148, pp. 151 – 156, June 2001.

[105] S. Kim and M. C. Papaefthymiou, "Single-phase source-coupled adiabatic logic," in *Proceedings of International Symposium on Low Power Electronics and Design, ISLPED'99*, Aug. 1999, pp. 97–99.

[106] S. Kim, C. H. Ziesler, and M. C. Papaefthymiou, "A true single-phase 8-bit adiabatic multiplier," in *Proceedings of the 38th ACM/IEEE Design Automation Conference*, June 2001, pp. 758–763.

[107] P. Ko, J. Huang, Z. Liu, and C. Hu, "BSIM3 for analog and digital circuit simulation," in *Proceedings of IEEE Symposium on VLSI technology CAD*, Jan 1993, pp. 400–429.

[108] U. Ko, P. T. Balsara, and W. Lee, "A self-timed method to minimize spurious transitions in low power CMOS circuits," in *Proceedings of IEEE Symposium on Low Power Electronics*, 1994, pp. 62–63.

[109] U. Ko, T. Balsara, and W. Lee, "Low-power design techniques for high-performance CMOS adders," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 3, no. 2, pp. 327–333, 1995.

[110] S. Komori, H. Takata, T. Tamura, F. Asai, T. Ohno, O. Tomisawa, T. Yamasaki, K. Shima, H. Nishikawa, and H. Terada, "A 40-MFLOPS 32-bit floating-point processor with elastic pipeline scheme," *IEEE Journal of Solid-State Circuits*, vol. 24, pp. 1341–1347, 1989.

[111] R. H. Krambeck, C. M. Lee, and H. F. S. Law, "High-speed compact circuits with CMOS," *IEEE Journal of Solid-State Circuits*, vol. SC-17, no. 3, pp. 614–619, 1982.

[112] R. H. Krambeck, C. M. Lee, and H. F. S. Law, "Low power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 4, pp. 473–484, 1992.

[113] A. Kramer, J. S. Denker, S. C. Avery, A. G. Dickinson, and T. R. Wik, "Adiabatic computing with the 2N-2N2D logic family," in *1994 Symposium on VLSI Circuits*, June 1994, pp. 25–26.

[114] N. Kumar, S. Katkoori, L. Rader, and R. Vemuri, "Profile-driven behavioral synthesis for low-power VLSI systems," *IEEE Design & Test of Computers*, vol. 12, no. 3, pp. 70–84, Fall 1995.

[115] T. Kuroda, T. Fujita, T. Nagamatsu, S. Yoshioka, T. Sei, K. Matsuo, Y. Hamura, T. Mori, M. Murota, M. Kakumu, and T. Sakurai, "A high-speed low power 0.3 μm CMOS gate array with variable threshold voltage (VT) scheme," in *Proceedings of IEEE 1996 Custom Integrated Circuit Conference, CICC'96*, May 1996, pp. 53–56.

[116] T. Kuroda and T. Sakurai, "Overview of low-power ULSI circuit techniques," *IEICE Transactions on Electronics*, vol. E78-C, pp. 334–344, 1995.

[117] T. Kuroda and T. Sakurai, "Threshold-voltage control schemes through substrate bias for low-power high-speed CMOS LSI design," *Journal of VLSI Signal Processing Systems*, vol. 13, no. 2/3, pp. 191–201, Aug. 1996.

[118] G. D. M. L. Benini, "System-level power optimization: Techniques and tools," *ACM Transactions on Design Automation of Electronic Systems*, vol. 5, no. 2, pp. 115–192, 2000.

[119] P. Landman and J. Rabaey, "Power estimation for high-level synthesis," in *Proceedings of IEEE European Conference on Design Automation, EDAC-93, Paris, France*, Feb 1993, pp. 361–366.

[120] P. Landman and J. Rabaey, "Activity-sensitive architectural power analysis for the control path," in *Proceedings of ACM/IEEE International Symposium on Low Power Design, ISLPD-95, Dana Point, CA*, April 1995, pp. 93–98.

[121] P.-K. Leung, C.-S. Choy, C.-F. Chan, and K.-P. Pun, "A low power asynchronous GF(2ˆ173) ALU for elliptic curve crypto-processor," in *Proceedings of the 2003 International Symposium on Circuits and Systems, ISCAS '03*, volume 5, May 2003, pp. 25–28.

[122] J. Lim, D. Kim, and S. Chae, "A 16-bit carry-lookahead adder using reversible energy recovery logic for Ultra-Low-Energy systems," *IEEE Journal of Solid-State Circuits*, vol. 34, no. 6, pp. 898–903, June 1999.

[123] R.-B. Lin and M.-C. Wu, "A new statistical approach to timing analysis of VLSI circuits," in *Proceedings of International Conference on VLSI Design*, 1998, pp. 507–513.

[124] J.-J. Liou, K.-T. Cheng, S. Kundu, and A. Krstic, "Fast statistical timing analysis by probabilistic event propagation," in *Proceedings of IEEE/ACM Design Automation Conference*, 2001, pp. 661–666.

[125] D. Liu and C. Svensson, "Power consumption estimation in CMOS VLSI chips," *IEEE Journal of Solid-State Circuits*, vol. 29, no. 6, pp. 663–670, 1994.

[126] Y. Lu and V. D. Agrawal, "Leakage and dynamic glitch power minimization using integer linear programming for Vth assignment and path balancing," in *Proceedings of the International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, 2005, pp. 217–226.

[127] D. Maksimović, V. Oklobdžija, B. Nikolić, and K. Current, "Clocked CMOS adiabatic logic with integrated single-phase power-clock supply: Experimental results," in *Proceedings of International Symposium on Low Power Electronics and Design, Monterey, CA*, 1997, pp. 323–327.

[128] M. Mani and M. Orshansky, "A new statistical optimization algorithm for gate sizing," in *Proceedings of IEEE International Conference on Computer Design*, 2004, pp. 272 – 277.

[129] D. Marculescu, R. Marculescu, and M. Pedram, "Information theorectic measures for power analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 6, pp. 599–610, 1996.

[130] R. Marculescu, D. Marculescu, and M. Pedram, "Logic level power estimation considering spatiotemporal correlations," in *Proceedings of the IEEE International Conference on Computer Aided Design*, Nov. 1994, pp. 294–299.

[131] R. Marculescu, D. Marculescu, and M. Pedram, "Efficient power estimation for highly correlated input streams," in *Proceedings of the 32nd Design Automation Conference*, June 1995, pp. 628–634.

[132] D. P. Marple, "Transistor size optimization in the tailor layout system," in *Proceedings of 26th ACM/IEEE Design Automation Conference*, June 1989, pp. 43–48.

[133] A. Marshall, B. Coates, and P. Siegel, "Designing an asynchronous communications chip," *IEEE Design & Test of Computer*, vol. 11, pp. 8–21, 1994.

[134] K. Martin, *Digital Integrated Circuit Design*, chapter 3, pp. 106–108. Oxford University Press, 2000.

[135] V. Mehrotra, S. Sam, D. Boning, A. Chandrakasan, R. Val1ishayee, and S. Nassif, "A methodology for modeling the effects of systematic within-die interconnect and device variation on circuit performance," in *Proceedings of the Design Automation Conference*, 2000, pp. 172 – 175.

[136] R. Merkle, "Reversible electronic logic using switches," *Nanotechnology*, vol. 4, pp. 21–40, 1993.

[137] J. Monteiro, S. Devadas, and A. Ghosh, "Estimation of switching activity in sequential logic circuits with applications to synthesis for low power," in *Proceedings of the 31st Design Automation Conference*, June. 1994, pp. 12–17.

[138] Y. Moon and D. K. Jeong, "An efficient charge recovery logic circuit," *IEEE Journal of Solid-State Circuits*, vol. 31, pp. 514–522, Apr. 1996.

[139] K. Muller-Glaser, K. Kirsch, and K. Neusinger, "Estimating essential design characteristics to support project planning for ASIC design management," in *Proceedings of IEEE/ACM International Conference on Computer Aided Design, ICCAD-91, Santa Clara, CA*, Nov. 1991, pp. 148–151.

[140] 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, 1995.

[141] L. W. Nagel, "SPICE2, a computer program to simulate semiconductor circuits," Technical Report ERL Memorandum ERL-M520, University of California, Electronics Research Laboratory, Berkeley, California, May 1975.

[142] F. N. Najm, "Transition density: a new measure of activity in digital circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 2, pp. 310–323, Feb. 1993.

[143] F. N. Najm, R. Burch, P. Yang, and I. N. Hajj, "CREST - a current estimator for CMOS circuits," in *Proceedings of IEEE International Conference on Computer-Aided Design*, Nov. 1988, pp. 204–207.

[144] F. N. Najm, R. Burch, P. Yang, and I. N. Hajj, "Probabilistic simulation for reliability analysis of CMOS VLSI circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 9, no. 4, pp. 439–450, April 1990.

[145] S. Nassif, "Delay variability: Sources, impacts and trends," in *IEEE International Solid-State Circuits Conference, ISSCC*, 2000, pp. 368–369.

[146] M. Nemani and F. Najm, "Towards a high-level power estimation capability," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 6, pp. 588–598, 1996.

[147] M. Nemani and F. Najm, "High-level area and power estimation for VLSI circuits," in *Proceedings of IEEE/ACM International Conference on Computer Aided Design, ICCAD-97, San Jose, CA*, Nov. 1997, pp. 114–119.

[148] M. Nemani and F. Najm, "High-level area prediction for power estimation," in *Proceedings of Custom Integrated Circuits Conference, CICC-97, Santa Clara, CA*, May 1997, pp. 483–486.

[149] L. S. Nielsen and J. Sparso, "An 85w asynchronous filterbank for a digital hearing aid." presented at International Solid-State Circuits Conference, Feb. 1998.

[150] L. S. Nielsen and J. Sparso, "Designing asynchronous circuits for low power: An IFIR filter bank for a digital hearing aid," *Proceedings of the IEEE*, vol. 87, no. 2, pp. 268–281, Fed. 1999.

[151] J. Oh and M. Pedram, "Gated clock routing minimizing the switched capacitance," in *Proceedings of Design Automation and Test in Europe Conference*, Feb. 1998, pp. 692–697.

[152] N. Ohkubo, M. Suzuki, T. Shinbo, T. Yamanaka, A. Shimizu, K. Sasaki, and Y. Nakagome, "A 4.4 ns CMOS 54×54-b multiplier using pass-transistor multiplexer," *IEEE Journal of Solid-State Circuits*, vol. 30, pp. 251–257, 1995.

[153] V. G. Oklobdžija, D. Maksimović, and F. C. Lin, "Pass-transistor adiabatic logic using single power-clock supply," *IEEE Transactions on Circuits and Systems II*, vol. 44, no. 10, pp. 842–846, Oct. 1997.

[154] M. Orshansky and K. Keutzer, "A general probabilistic framework for worst-case timing analysis," in *Proceedings of Design Automation Conference*, 2002, pp. 556–569.

[155] M. Orshansky, L. Milor, P. Chen, K. Keutzer, and C. Hu, "Impact of systematic spatial intra-chip gate length variability on performance of high-speed digital circuits," in *Proceedings of IEEE International Conference on Computer-Aided Design*, 2000, pp. 62–67.

[156] W.-H. Paik, H.-J. Ki, and S.-W. Kim, "Push-pull pass-transistor logic family for low-voltage and low-power," in *Proceedings of 22nd Europe Solid-State Circuits Conference, Neuchâtel, Switzerland*, Sept. 1996, pp. 116–119.

[157] P. R. Panda, F. Catthor, N. D. Dutt, K. Danckaert, E. Brockmeyer, C. Kulkami, A. Vandercappele, and P. G. Kjeldsberg, "Data and memory optimization techniques for embedded systems," *ACM Transactions on Design Automation of Electronic Systems*, vol. 6, no. 2, pp. 149–206, April 2001.

[158] A. Parameswar, H. Hara, and T. Sakurai, "A swing restored passtransistor logic-based multiply and accumulate circuit for multimedia applications," *IEEE Journal of Solid-State Circuits*, vol. 31, pp. 804–809, 1996.

[159] J. H. Pasternak and C. A. T. Salama, "Differential pass-transistor logic," *IEEE Circuits & Devices*, pp. 23–28, July 1993.

[160] M. Pattanaik, S. Banerjee, and B. Bahinipati, "GP based transistor sizing for optimal design of nanoscale CMOS inverter," in *Proceedings of the Third IEEE Conference on Nanotechnology (IEEE-NANO 2003)*, volume 2, Aug. 2003, pp. 524 – 527.

[161] N. Paver and D. Edwards, "Is asynchronous logic good for low-power?," in *IEE Colloquium on Low Power Analogue and Digital VLSI: ASICS, Techniques and Applications*, Jun. 1995, pp. 4/1 – 4/5.

[162] N. C. Paver, P. Day, C. Farnsworth, D. L. Jackson, W. A. Lien, and J. Liu, "A low-power, low-noise configurable self-timed DSP," in *Proceedings of International Symposium on Advanced Research in Asynchronous Circuits and Systems*, 1998, pp. 32–42.

[163] N. C. Paver and S. B. Furber, "AMULET2: Increasing power efficiency," in *Proceedings of 1995 Israel Workshop on Asynchronous VLSI*, March 1995, pp. 19–22.

[164] M. Pedram and J. Rabaey, *Power-aware design methodologies*. Kluwer Academic Publishers, 2002.

[165] N. R. Poole, "Self timed logic circuits," *IEE Electronics & Communication Engineering Journal*, vol. 6, no. 6, pp. 261–270, 1994.

[166] S. Powell and P. Chau, "Estimating power dissipation of VLSI signal processing chips: The PFA techniques," in *Proceedings of IEEE Workshop on VLSI Signal Processing*, volume VI, 1990, pp. 250–259.

[167] J. Quintana, M. Avedillo, R. Jimenez, and E. Rodriguez-Villegas, "Low-power logic styles for full-adder circuits," in *Proceedings of the 8th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2001*, volume 3, Sept. 2001, pp. 1417 – 1420.

[168] J. M. Rabaey and M. Pedram, *Low power design methodologies*. Boston: Kluwer Academic Publishers, 1996.

[169] T. Raja, *Minimum dynamic power CMOS design with variable input delay logic*. PhD thesis, Rutgers University, 2004.

[170] T. Raja, V. D. Agrawal, and M. L. Bushnell, "CMOS circuit design by a reduced constraint set linear program," in *Proceedings of the International Conference on VLSI Design*, Jan. 2003, pp. 527–532.

[171] T. Raja, V. D. Agrawal, and M. L. Bushnell, "CMOS circuit design for minimum dynamic power and highest speed," in *Proceedings of the International Conference on VLSI Design*, Jan. 2004, pp. 1035–1040.

[172] T. Raja, V. D. Agrawal, and M. L. Bushnell, "Design of variable input delay gates for low dynamic power circuits," in *Proceedings of 15th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS 2005)*, Sept. 2005, pp. 436–445.

[173] T. Raja, V. D. Agrawal, and M. L. Bushnell, "Variable input delay CMOS logic for low power design," in *Proceedings of the International Conference on VLSI Design*, Jan. 2005, pp. 596–604.

[174] K. Roy and S. C. Prasad, *Low-Power CMOS VLSI Circuit Design*. New York: Wiley Interscience Publication, 2000.

[175] J. Rubinstein, P. Penfield, and M. A. Horowitz, "Signal delay in RC tree networks," *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, vol. CAD-2, pp. 202–211, 1983.

[176] H. Sakamoto, H. Ochi, K. Uda, K. Taki, and B.-Y. L. T. Tsuda, "A 16-bit redundant binary multiplier using low-power pass-transistor logic SPL," in *Proceedings of Design Automation Conference, Asia and South Pacific (ASP-DAC 2000)*, Jan. 2000, pp. 33–34.

[177] A. Salz and M. A. Horowitz, "IRSIM: An incremental MOS switch-level simulator," in *Proceedings of the 26th Design Automation Conference*, June 1989, pp. 173–178.

[178] S. S. Sapatnekar, V. B. Rao, P. M. Vaidya, and S. M. Kang, "An exact solution to the transistor sizing problem for CMOS circuits using convex optimization," *IEEE Transactions on Computer-Aided Design*, vol. 12, pp. 1621–1634, Nov. 1993.

[179] C. V. Schimpfle, A. Wroblewski, and J. A. Nassek, "Transistor sizing for switching activity reduction in digital circuits," in *Proceedings of the European Conference on Theory and Design*, volume 1, Aug. 1999, pp. 114–117.

[180] P. Schneider and U. Schlichtmann, "Decomposition of boolean functions for low power based on a new power estimation technique," in *Proceedings of the 1994 International Workshop on Low Power Design*, April 1994, pp. 123–128.

[181] K. Schuegraf and C. Hu, "Hole injection SiO2 breakdown model for very low voltage lifetime extrapolation," *IEEE Transactions on Electronic Devices*, vol. 41, pp. 761–767, May 1994.

[182] C. L. Seitz, *Introduction to VLSI systems*, chapter 7, pp. 218–262. Reading, MA:Addison-Wesley, 1980.

[183] K. Seta, H. Hara, T. Kuroda, M. Kakumu, and T. Sakurai, "50% active-power saving without speed degradation using standby power reduction (SPR) circuit," in *IEEE International Solid-State Circuits Conference Digest of Technical papers*, Feb. 1995, pp. 318–319.

[184] J. M. Shyu, A. L. Sangiovanni-Vincntelli, J. P. Fishburn, and A. E. Dunlop, "Optimization-based transistor sizing," *IEEE Journal of Solid-State Circuits*, vol. 23, no. 2, pp. 400–409, Apr. 1988.

[185] P.-L. Siu, C.-S. Choy, J. Butas, and C. Chan, "A low power asynchronous DES," in *Proceedings of the 2001 IEEE International Symposium on Circuits and Systems, ISCAS 2001*, volume 4, May 2001, pp. 538–541.

[186] D. Somasekhar, Y. Ye, and K. Roy, "A energy recovery static RAM memory core," in *Proceedings of International Symposium on Low-Power Electronics and Design*, 1995, pp. 62–63.

[187] M. Song, G. Kang, S. Kim, and B. Kang, "Design methodology for high speed and low power digital circuits with energy economized passtransistor logic (EEPL)," in *Proceedings of 22nd Europe Solid-State Circuits Conference, Neuchâtel, Switzerland*, Sept. 1996, pp. 120–123.

[188] A. Srivastava and D. Sylvester, "A general framework for probabilistic low-power design space exploration considering process variation," in *Proceedings of IEEE/ACM International Conference on Computer Aided Design*, Nov. 2004, pp. 808 – 813.

[189] H. Stark and J. W. Woods, *Probability, random processes, and estimation theory for engineers.* Prentice-hall, 1986.

[190] V. Sundararajan, S. Sapatnekar, and K. Parhi, "Fast and exact transistor sizing based on iterative relaxation," *IEEE Transactions on Computer Aided Design of Circuits and Systems*, vol. 21, no. 5, pp. 568–581, 2002.

[191] I. Sutherland, "Micropipelines," *Communication of ACM*, vol. 32, no. 6, pp. 720–738, 1989.

[192] S. Sze, *Physics of semiconductor devices.* John Wiley & Sons, 1981.

[193] K. Taki and B.-Y. Lee, "Low power pass-transistor logic and application examples," *IEICE Transactions on Electronics, Information and Communication*, vol. J80-A, no. 5, pp. 1–12, 1997. (in Japanese).

[194] Y. Taur and T. H. Ning, *Fundamentals of Modern VLSI Devices*, chapter 2, pp. 94–97. New York, USA: Cambridge University Press, 1998.

[195] H. Terada, M. Iwata, S. Miyata, and S. Komori, "Superpipelined dynamic data-driven VLSI processors," in *Advanced Topics in Dataflow Computing and Multithreading*, (Los Alamitos, CA), IEEE Computer Society Press, 1995, pp. 75–85.

[196] H. Terada, S. Miyata, and M. Iwata, "DDMP's: Self-timed super-pipelined data-driven multimedia processors," *Proceedings of the IEEE*, vol. 87, no. 2, pp. 282–296, 1999.

[197] G. Theodoridis, S. Theoharis, D. Soudris, T. Stouraitis, and C. Goutis, "An efficient probabilistic method for logic circuits using real delay gate model," in *the 1999 IEEE International Symposium on Circuits and Systems, ISCAS '99*, volume 1, Jun 1999, pp. 286–289.

[198] S. Thompson, I. Young, J. Greason, and M. Bohr, "Dual threshold voltages and substrate bias: Keys to high performance, low power, 0.1 μm logic designs," in *1997 Symposium on VLSI Technology, Digest of Technical Papers*, 1997, pp. 69–70.

[199] Y.-F. Tsai, D. Duarte, N. Vijaykrishnan, and M. J. Irwin, "Implications of technology scaling on leakage reduction techniques," in *Proceedings of the Design Automation Conference*, 2003, pp. 187–190.

[200] Y.-F. Tsai, N. Vijaykrishnan, Y. Xie, and M. Irwin, "Influence of leakage reduction techniques on delay/leakage uncertainty," in *Proceedings of the International Conference on VLSI Design*, Jan 2005, pp. 374 – 379.

[201] C.-Y. Tsui, M. Pedram, and A. M. Despain, "Efficient estimation of dynamic power consumption under a real delay model," in *Proceedings of IEEE International Conference on Computer-Aided Design, Santa Clara, CA*, Nov. 1993, pp. 224–228.

[202] C.-Y. Tsui, M. Pedram, and A. M. Despain, "Exact and approximate methods for calculating signal and transition probabilities in FSMS," in *Proceedings of the 31st Design Automation Conference*, June 1994, pp. 18–23.

[203] N. Tzartzanis and W. C. Athas, "Energy recovery for the design of high-speed, low-power static RAM," in *Proceedings of International Symposium on Low-Power Electronics and Design*, 1996, pp. 55–60.

[204] S. Uppalapati, "Low power design of standard cell digital VLSI circuits," Master's thesis, Rutgers University, New Brunswick, New Jersey, Oct. 2004.

[205] S. Uppalapati, M. L. Bushnell, and V. D. Agrawal, "Glitch-free design of low power ASICS using customized resistive feedthrough cells," in *Proceedings of the 9th VLSI Design and Test Symposium*, 2005, pp. 41–48.

[206] M. M. Vai, *VLSI Design*. Boca Raton, Florida: CRC press LLC, 2001.

[207] K. van Berkel, R. Burgess, J. Kessels, A. Peeters, M. Roncken, and F. Schalij, "Asynchronous circuits for low power: A DCC error corrector," *IEEE Design & Test of Computers*, vol. 11, no. 2, pp. 22–32, 1994.

[208] K. van Berkel, R. Burgess, J. Kessels, A. Peeters, M. Roncken, F. Schalij, and R. van de Wiel, "A single-rail re-implementation of a DCC error detector using a generic standard-cell library," in *Proceedings of Asynchronous Design Methodologies (Async'95), London, UK*, 1995, pp. 72–79.

[209] K. van Berkel, J. Kessels, M. Roncken, R. Saeijs, and F. Schalij, "The VLSI-programming language Tangram and its translation into handshake circuits," in *Proceedings of European Conference on Design Automation (EDAC'91)*, 1991, pp. 384–389.

[210] H. van Gageldonk, D. Baumann, K. van Berkel, D. Gloor, A. Peeters, and G. Stegmann, "An asynchronous low-power 80c51 microcontroller," in *Proceedings of International Symposium on Advanced Research in Asynchronous Circuits and Systems*, 1998, pp. 96–107.

[211] H. J. M. Veendrick, "Short-circuit dissipation of static CMOS circuitry and its impact on the design of buffer circuits," *IEEE Journal of Solid-State Circuits*, vol. SC-19, pp. 468–473, 1984.

[212] C. Vieru, "Pendulum: A reversible computer architecture," Master's thesis, Massachusetts Institute of Technology, 1995.

[213] V. Von Kaenel, P. Macken, and M. Degrauwe, "A voltage reduction technique for battery-operated systems," *IEEE Journal on Solid-State Circuits*, vol. 25, no. 5, pp. 1136–1140, 1990.

[214] F. M. Wanlass and C. T. Sah, "Nanowatt logic using field effect metal-oxide semiconductor triode," in *IEEE International Solid State Conference, Digest of Technical Papers*, volume VI, Feb. 1963, pp. 32–33.

[215] L. Wei, Z. Chen, K. Roy, M. Johnson, Y. Ye, and V. De, "Design and optimization of dual-threshold circuits for low-voltage low-power applications," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 7, no. 1, pp. 16–24, Mar 1999.

[216] E. W. Weisstein, "Linear programming." MathWorld – A Wolfram Web Resource. Available at http://mathworld.wolfram.com/LinearProgramming.html, 2005.

[217] J. Wood, T. Edwards, and S. Lipa, "Rotary travelling-wave oscillator arrays: A new clock technology," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 11, pp. 1654–1665, Nov. 2001.

[218] M. K. Wood and G. B. Dantzig, "Programming of interdependent activities. I. general discussion," *Econometrica*, vol. 17, pp. 193–199, 1949.

[219] A. Wróblewski, C. Schimpfle, O. Schumacher, and J. A. Nossek, "Minimizing spurious switching activities with transistor sizing," *Journal on VLSI Design*, vol. 15, no. 2, pp. 537–546, Sept. 2002.

[220] A. Wróblewski, C. V. Schimpfle, and J. A. Nassek, "Automated transistor sizing algorithm for minimizing spurious switching activities in CMOS circuits," in *Proceedings of the IEEE International Symposium on Circuits and Systems*, volume 3, May 2000, pp. 291–294.

[221] M. Xakellis and F. Najm, "Statistical estimation of the switching activity in digital circuits," in *Proceedings of 31st ACM/IEEE Design Automation Conference, San Diego, CA*, 1994, pp. 728–733.

[222] G. Y. Yacoub and W. H. Ku, "An accurate simulation technique for short circuit power dissipation based on current component isolation," in *Proceedings of the IEEE International Symposium on Circuit and Systems*, 1989, pp. 1157–1161.

[223] M. Yamada, S. Kurosawa, R. Nojima, N. Kojima, T. Mitsuhashi, and N. Goto, "Synergistic power/area optimization with transistor sizing and wire length minimization," in *1994 IEEE Symposium on Low Power Electronics, Digest of Technical Papers*, 1994, pp. 50–51.

[224] S. Yamashita, K. Yano, Y. Sasaki, Y. Akita, H. Chikata, K. Rikino, and K. Seki, "Pass-Transistor/CMOS collaborated logic: The best of both worlds," in *Symposium on VLSI Circuits, Digest of Technical Papers*, 1997, pp. 31–32.

[225] C. Yang and M. Ciesielski, "Synthesis for mixed CMOS/PTL logic: Preliminary results," in *International Workshop on Logic Synthesis, Lake Tahoe, CA*, 1999.

[226] N. Yang, W. K. Henson, and J. Wortman, "A comparative study of gate direct tunneling and drain leakage current in N-MOSFETs with sub-2100-nm gate oxides," *IEEE Transactions on Electronic Devices*, vol. 47, pp. 1636–1644, 2000.

[227] K. Yano, Y. Sasaki, K. Rikino, and K. Seki, "Top-down pass-transistor logic design," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 6, pp. 792–803, 1996.

[228] K. Yano, T. Yamanaka, T. Nishida, M. Saito, K. Shimohigashi, and A. Shimizu, "A 3.8ns CMOS 16×16-b multiplier using complementary pass-transistor logic," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 2, pp. 388–395, 1990.

[229] Y. Ye, S. Borkar, and V. De, "A new technique for standby leakage reduction in high-performance circuits," in *Proceedings of the 1998 Symposium on VLSI Circuits*, 1998, pp. 40–41.

[230] Y. Ye and K. Roy, "QSERL: Quasi-static energy recovery logic," *IEEE Journal of Solid-State Circuits*, vol. 36, no. 2, pp. 239–248, Feb. 2001.

[231] S. Younis, *Asymptotically zero energy computing using split-level charge recovery logic.* PhD thesis, Massachusetts Institute of Technology, Cambridge, June 1994.

[232] S. G. Younis and T. F. Knight, "Asymptotically zero energy split-level charge recovery logic," in *Proceedings of International Workshop on Low Power Design, Napa Valley, CA*, 1994, pp. 177–182.

[233] C. Ziesler, J. Kim, and M. Papaefthymiou, "Energy recovering ASIC design," in *Proceedings of IEEE Computer Society Annual Symposium on VLSI*, Feb. 2003, pp. 133 – 138.

[234] R. Zimmermann and W. Fichtner, "Low-power logic styles: CMOS versus pass-transistor logic," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 7, pp. 1079–1090, 1997.