Architectural Level Delay and Leakage Power Modeling of Manufacturing Process Variation

A Thesis Submitted for the Degree of Doctor of Philosophy in the

Faculty of Engineering

by

Chenxi Ni

School of Electrical, Electronic and Computer Engineering
Newcastle University
Newcastle upon Tyne
United Kingdom

September-2012

ABSTRACT

The effect of manufacturing process variations has become a major issue regarding the estimation of circuit delay and power dissipation, and will gain more importance in the future as device scaling continues in order to satisfy market place demands for circuits with greater performance and functionality per unit area. Statistical modelling and analysis approaches have been widely used to reflect the effects of a variety of variational process parameters on system performance factor which will be described as probability density functions (PDFs). At present most of the investigations into statistical models has been limited to small circuits such as a logic gate. However, the massive size of present day electronic systems precludes the use of design techniques which consider a system to comprise these basic gates, as this level of design is very inefficient and error prone.

This thesis proposes a methodology to bring the effects of process variation from transistor level up to architectural level in terms of circuit delay and leakage power dissipation. Using a first order canonical model and statistical analysis approach, a statistical cell library has been built which comprises not only the basic gate cell models, but also more complex functional blocks such as registers, FIFOs, counters, ALUs etc. Furthermore, other sensitive factors to the overall system performance, such as input signal slope, output load capacitance, different signal switching cases and transition types are also taken into account for each cell in the library, which makes it adaptive to an incremental circuit design.

The proposed methodology enables an efficient analysis of process variation effects on system performance with significantly reduced computation time compared to the Monte Carlo simulation approach. As a demonstration vehicle for this technique, the delay and leakage power distributions of a 2-stage asynchronous micropipeline circuit has been simulated using this cell library. The experimental results show that the proposed method can predict the delay and leakage power distribution with less than 5% error and at least 50,000 times faster computation time compare to 5000-sample SPICE based Monte Carlo simulation.

The methodology presented here for modelling process variability plays a significant role in Design for Manufacturability (DFM) by quantifying the direct impact of process variations on system performance. The advantages of being able to undertake this analysis at a high level of abstraction and thus early in the design cycle are two fold. First, if the predicted effects of process variation render the circuit performance to be outwith specification, design modifications can be readily incorporated to rectify the situation. Second, knowing what the acceptable limits of process variation are to maintain design performance within its specification, informed choices can be made regarding the implementation technology and manufacturer selected to fabricate the design.

LIST OF PUBLICATIONS

Conferences Paper

[1] **Chenxi Ni**, Z.AL Tarawneh, G. Russell and A. Bystrov, “Statistical delay and leakage power modeling of manufacturing process variation at architectural level,” *proceeding* to *IEEE international Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, April, 2012.

[2] **Chenxi Ni**, Gordon Russell and Alex bystrov, “Statistical delay modeling of manufacturing process variation at system level,” in *proceeding of IEEE international Northeast Workshop on Circuits and Systems (NEWCAS),* pp.133-136, 2012.

[3] **Chenxi Ni**, Z.AL Tarawneh, G. Russell and A. Bystrov, “Statistical leakage power modeling of manufacturing process variation at system level,” in *IEEE Power Engineering and Automation Conference*, 2012.

Submitted Journal Paper

[1] **Chenxi Ni**, Z.AL Tarawneh, G. Russell and A. Bystrov, “Architectural level modeling of manufacturing process variation,” submitted to *IET Computer & Digital Techniques Journal (CDT),* Dec 2011.

Submitted Journal Paper

[1] **Chenxi Ni**, Z.AL Tarawneh, G. Russell and A. Bystrov, “System level modeling of process variation effects on circuit performance,” in *UK Electronic Forum*, 2012.

[2] **Chenxi Ni**, Z.AL Tarawneh, G. Russell and A. Bystrov, “System level modeling of manufacturing process variation,” in *Post Graduate Conference,* Newcastle University, 2010.

Table of Contents

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

1.1 Background

In order to satisfy the market place demand for circuits with greater performance and functionality per unit area, the semiconductor and fabrication technology has been rapidly developed during the last century, which has lead to the continuous shrinking of transistor dimensions. According to Moore's Law [1], the number of transistors that can be fabricated on an integrated circuit doubles approximately every two years. This trend has continued for more than half a century, and is expected to continue until at least 2015 or 2020 [2]. Figure 1-1 shows the plot of CPU transistor counts against dates from 1971 to 2011.

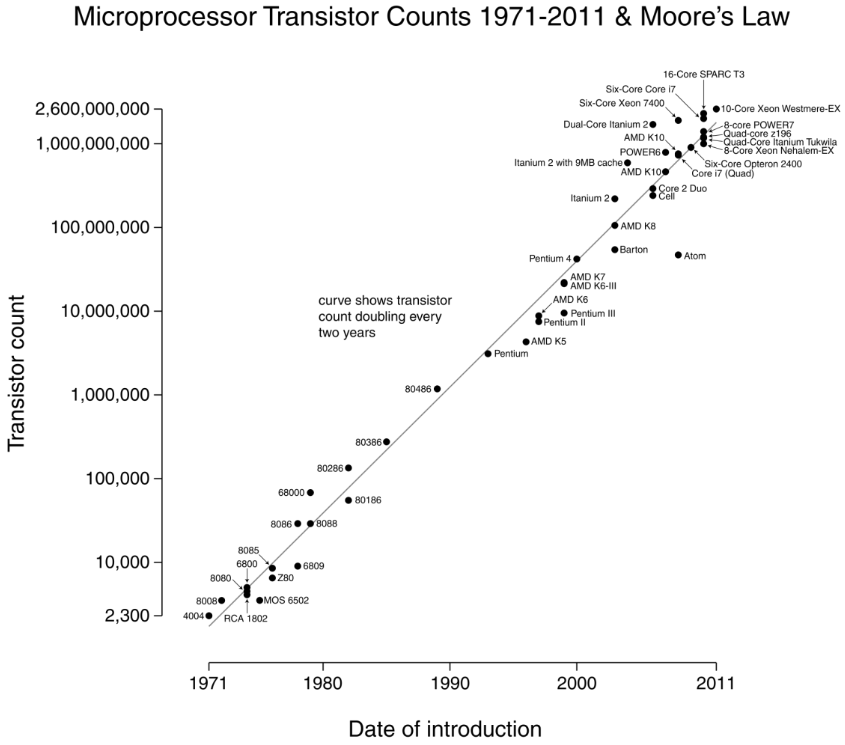


Figure 1-1 Plot of CPU transistor counts against dates of introduction [3].

As a result of technology scaling and the increase in circuit density, process and environmental variation effects have been highlighted as the main reasons for the uncertainty in circuit behavior. Electronic systems are becoming more susceptible to these variations which not only impact on system performance but also on system reliability. System reliability issues are of growing concern due to the range of applications in which electronic systems are used, for example in automotive, aerospace and medical applications.

In this chapter, a general introduction to the manufacturing process variations will be outlined; the subsequent sections are organized as follows: Section 1.2 will give a general introduction to impact of device scaling. Section 0 will present the classification of the variation sources and Section 1.4 will illustrate the different components of these variations. In Section 1.5 the impact of variation on performance parameters will be discussed; followed by the motivation and contributions of this thesis in Section 1.6; the chapter will conclude with a description of the roadmap of the thesis in Section 1.7.

1.2 Impact of Device Scaling

In order to satisfy the market demand for high performance electronic systems, the circuit implementation requires the transistor density per unit area to be as high as possible. This results in a chip with the same functionality occupying a smaller area, or a chip in the same area with more functionality. Since the costs for fabricating a semiconductor wafer are relatively fixed, then the cost of an integrated circuit (IC) is mainly related to the number of chips which can be packed on a wafer. Hence, smaller ICs allow more chips on a wafer, reducing the price per chip.

In fact, the number of transistors per chip has been doubled every 2-3 years following the Moore’s law during last 3 decades as described in section 1.1. The transistor dimensions have been scaled down dramatically which keep pushing the semiconductor technology to new nodes. The scaling of the MOSFET dimensions into the Deep Sub-Micro (DSM) regime gives significant improvement in system performance, and will continue progressively in the future according to the International Technology Roadmap for Semiconductors (ITRS) [4]. Figure 1-2 shows an example which compares the physical size of the old and modern electronic devices. In Figure 1-2, the larger device is an Osborne Executive portable computer from 1982 with a Zilog Z80 4MHz CPU; and the smaller one is an Apple iPhone 3GS with a 412MHz ARM11 CPU which released in 2009. The new iPhone is almost 100 times lighter in weight and 500 times smaller in volume than the old Osborne Executive computer. On the other hand, it is also, at least, 100 times faster and 10 times cheaper. Obviously, electronic systems definitely benefit from the device scaling.

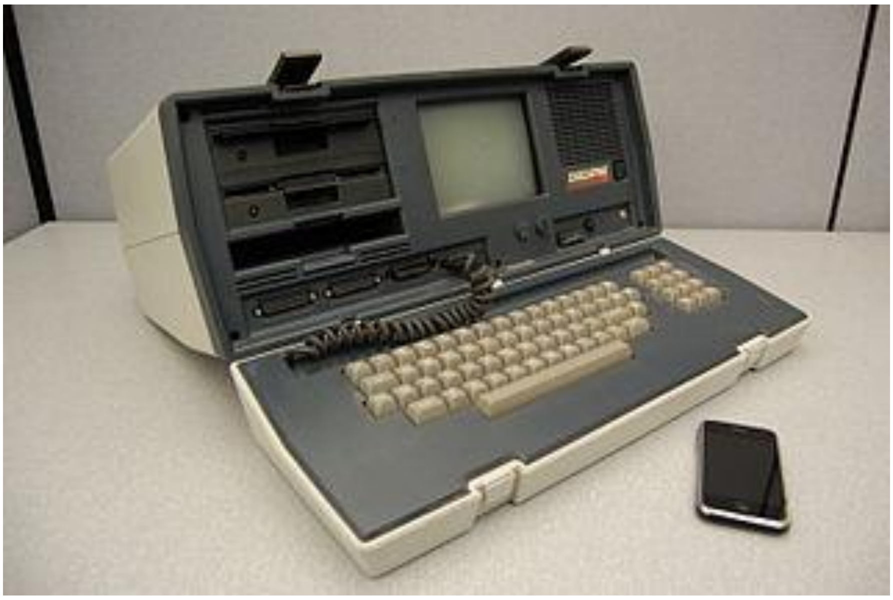


Figure 1-2 Size difference between old computers and modern ones.

In general, there are 2 basic type of scaling: *constant-field scaling* and *constant-voltage scaling*. In *constant-field scaling* (also referred to *full scaling*), the internal electric field in devices are preserved when the physical dimensions such as gate length L, width W and oxide-thickness T_{ox} scale down by a factor S. To maintain the constant field the voltage V_{GS}, V_{DS} and V_T must also be scaled down. Furthermore, the substrate doping must also be scaled to maintain the internal electric field of devices. However, because of the external voltage-level constrains, the *constant-field scaling* is not practical. As a consequence *constant-voltage scaling* has been preferred. In *constant-voltage scaling*, all dimensions are reduced by a factor S, the power supply and terminal voltages remaining unchanged. *Constant-voltage scaling* can provide voltage compatibility with older circuit technologies but enhance the internal electric field, which can cause a lot of reliability problems. Based on the literature [5], the *constant-voltage scaling* increases the device power density and drive current density by a factor of S^3. Such a huge increase in power and current density may eventually lead to hot-carrier degradation, electro-migration, and oxide breakdown etc. Table 1-1 lists the scaling factors of the main MOSFET device parameters for *constant-field scaling* and *constant-voltage scaling*. Table 1-2 compares the effects of *constant-field scaling* and *constant-voltage scaling* upon key MOSFET device characteristics [5].

Table 1-1: Comparison of constant-field scaling and constant-voltage scaling of main device parameters[5].

Device Parameter	Symbol	Constant-Field Scaling	Constant-Voltage Scaling
Channel length	L	$1/S$	$1/S$
Channel width	W	$1/S$	$1/S$
Oxide thickness	T_{ox}	$1/S$	$1/S$
Junction depth	X_j	$1/S$	$1/S$
Supply voltage	V_{dd}	$1/S$	1
Threshold voltage	V_{th}	$1/S$	1
Doping densities	N_A,N_D	S	S^2

Table 1-2 Comparison of effect of constant-field scaling and constant-voltage scaling upon key MOSFET device characteristics[5].

Device Parameter	Symbol	Constant-Field Scaling	Constant-Voltage Scaling
Oxide capacitance	C_{ox}	S	S
Drain current	I_d	$1/S$	S
Power dissipation	P	$1/S^2$	S
Power density	$P/Area$	1	S^3

1.3 Source of Variations

A robust circuit design ensures that the estimated performance deviation is within the limits of an acceptable yield. However, the variability introduced during a process will lead to the fluctuations in the values of system behavioural parameters, such as delay and power. The difficult part of performance prediction is that the fluctuation does not result from a single variation source. In order to study the impact of variation during the design process, the first step is to distinguish different variation phenomena. Figure 1-3 illustrates three types of variations introduced during the corresponding steps in the design of a system.

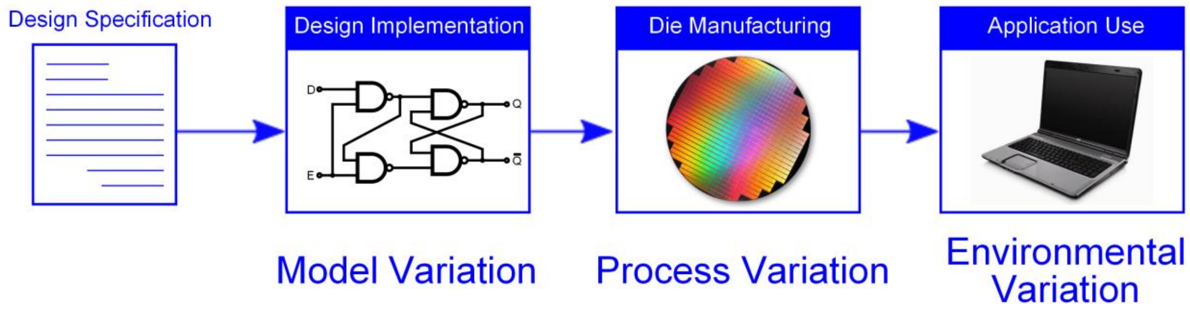


Figure 1-3 Steps of the design process and their resulting variations [6].

1.3.1 Model Variation

Modelling variation is mainly caused by the fact that the delay and power models, such as SPICE model, cannot perfectly capture the characteristics of the devices during the design analysis and optimization procedures. The inaccuracy of these models will result in a deviation between the predicted system performance and its expected performance in terms of delay and power dissipation. The aggressive models which lead to an under estimated prediction will cause yield loss. The conservative models which lead to an over estimated prediction will make it harder to meet design specifications, however, these models, typically, guarantee that the system performance is within a certain range of specifications.

1.3.2 Process Variation

Process variations result from a wide range of factors during fabrication such as threshold voltage adjustment implantation energy, High-k dielectric thickness, substrate doping etc. The fluctuation in the values of the uncontrollable fabrication parameters leads to the deviation of device parameters with respect to their expected value. These variations will affect the device after fabrication no matter the operating conditions. With the semiconductor technology shrinking into the nanometer range, the resulting process variability due to fabrication parameters or statistical variations of a small number of dopants, becomes increasingly important [4, 7, 8]. Some performance-sensitive device parameters such as transistor effective length (L_{eff}), width (W), oxide thickness (T_{ox}) and threshold voltage (V_{th}) show a significant amount of variation in nanometer regions [4, 9, 10]. The consequence of larger variations is that the device characteristics deviate strongly from its expected values. These effects will spread across the whole die and cause an undesirable spread of system performance affecting the parametric yield, which is defined as the percentage of dies that satisfy specific frequency and power constrains [8], a significant yield-loss will increase the unit cost of the product.

1.3.3 Environmental Variations

The environmental variations comprise the variations in switching activity which is defined by the input vectors, the variation of supply voltage (V_{dd}) and the variation of the operating temperature (T). Integrated circuit designs require the devices to work within a specific temperature range, because the increase of temperature will degrade the system performance. The typical solution to this problem is to use a lower supply voltage. However, the reduced V_{dd} will not only decrease the device driving strength hence degrade the system performance again, but also enhance the variation effects of the environmental factors [9].

The supply voltage is usually suffering a power drop off caused by the leakage current flow in devices even when the circuit is in the stand-by state; this effect could be neglected before nanometer technology. However, with the continuous shrinking of transistor dimensions and supply voltage, the fluctuation magnitude of leakage current is becoming larger. As a consequence, the variation of supply voltage becomes more significant with respect to circuit performance. At the same time, the operating temperature is also showing a large amount of variation since this factor is highly related to V_{dd} and leakage current. Interestingly, the leakage currents themselves also increase strongly with an increase in temperature, just as increasing leakage currents may result in a higher temperature [11], which brings more uncertainty to the overall circuit performance. Unlike process variations, the environmental variations depend on the work-load of the processor and are hence time-dependent. Therefore these variation sources can only temporarily affect the system performance [12] and circuit failures occur only intermittently during its operational life time [9].

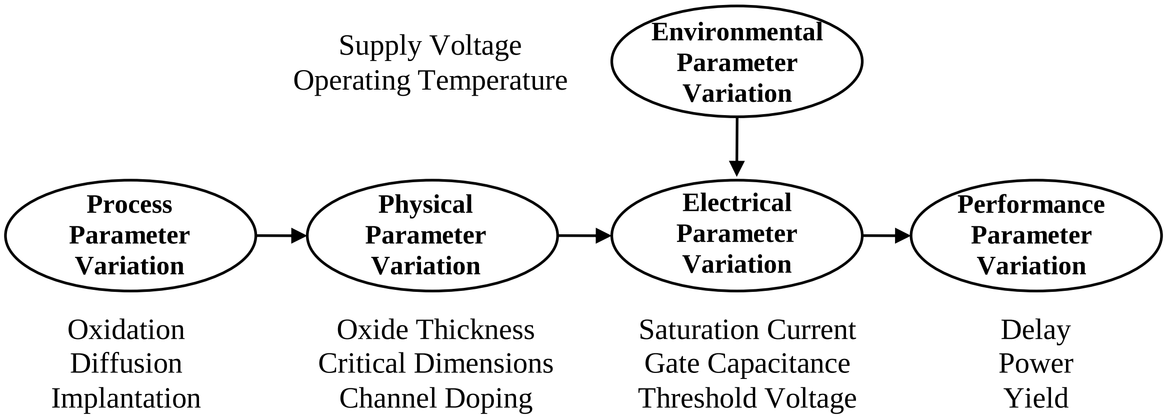


Figure 1-4 The propagation of the variation effects.

As shown in Figure 1-4, the uncertainties of both the operating environments and the device physical parameters which are caused by the process variation, will lead to the fluctuation of the electrical parameters such as transistor saturation current, gate capacitance, threshold voltage, etc. Subsequently, the fluctuations of electrical parameters will result in the variation of the circuit performance in terms of delay, power and yield.

1.3.4 Other Sources of Variation

The categories described above cover the majority of the sources of variation, however, there are also other sources which introduce the uncertainty into circuits with time. Negative Bias Temperature Instability (NBTI) and the Hot Carrier Injection (HCI) phenomena are the key reliability issues for MOSFET transistors. Their effects will result in an increase in the transistor threshold voltage, which leads to the device performance degradation even failure [13, 14]. On the other hand, the interconnect also suffers a negative impact from the electromigration phenomenon. This effect will cause a reduction in the width of wires, thus increase its resistance, resulting in an open circuit in the worst case [15]. These time dependent sources of variation are closely associated with the fabrication environment, and the effects will only become apparent in the field some time in the future. Therefore these effects are extremely difficult to model and analyze. Techniques such as burn-in can be used to test device reliability by accelerating their life time and detecting early-life failures. However, this kind of testing approach is very expensive and time consuming.

1.4 Components of Process Variation

For design analysis purposes, the components of process variation have to be studied first since they will influence the circuit performance differently. The general taxonomy of process variations is shown in Figure 1-5.

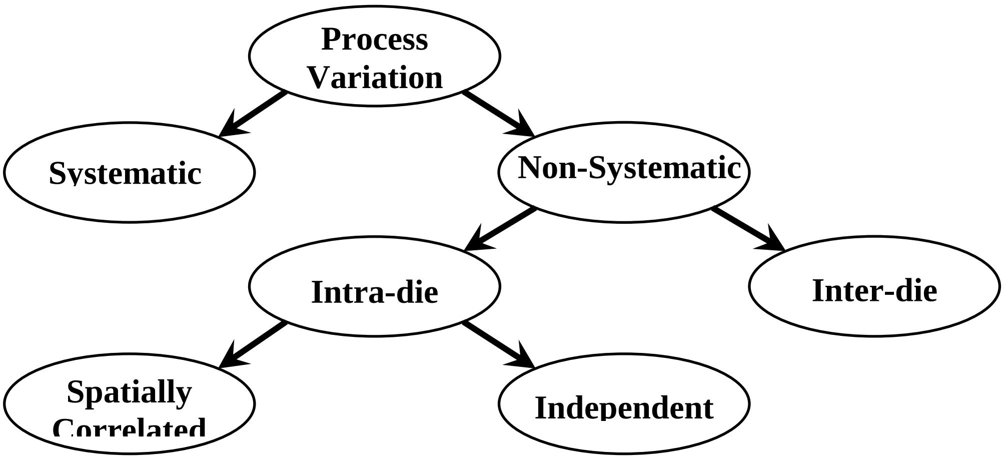


Figure 1-5 Taxonomy of process variations [6].

1.4.1 Systematic and Non-Systematic Variations

In general, sources of process variation can be classified into 2 groups based on whether they are deterministic or truly random, and are referred to as systematic and non-systematic variations respectively.

1) The systematic variations follow a known behaviour with the system layout. This kind of variation can be introduced during a number of steps in the manufacturing process. These include optical lithography (Photolithography) which is used to selectively remove parts of a thin film or the bulk of a substrate, the chemical

mechanical polishing (CMP) which is used to planarize insulating oxides and metal lines, and the associated metal fill which is typically added to design data during chip finishing just before tape-out [16-18]. The systematic variations are layout-dependent and can be modelled pre-manufacturing using a full layout analysis, thus this kind of variation effect can be predicted at the later stage of design cycle [19, 20]. However, since the layout information and the models required for analysis of the systematic variations are normally unavailable to the designer at the beginning of design process, it commonly treats these variations statistically.

2) The non-systematic variation is also known as random variation, which represent the true uncertain components of process variation. This kind of uncertainty cannot be predicted deterministically, and only the statistical characteristics are known at design time. Examples of sources of non-systematic variation include the line edge roughness (LER) which describes the uniformity of a single line along a limited length [21], and the random dopant fluctuations (RDF) which is a form of process variation resulting from variation in the implanted impurity concentration [22].

Both systematic and non-systematic variations are commonly assumed as random quantities at the early stage of the design process. When the design process moves to the next stage, the detail layout information will be obtained. If design analysis capability allow, the systematic variation can be modeled deterministically, thus the overall variability of the design will be reduced.

1.4.2 Inter-Die and Intra-Die Variations

The non-systematic variations can be further classified into 2 categories based on how the sources of variation act on different spatial scales.

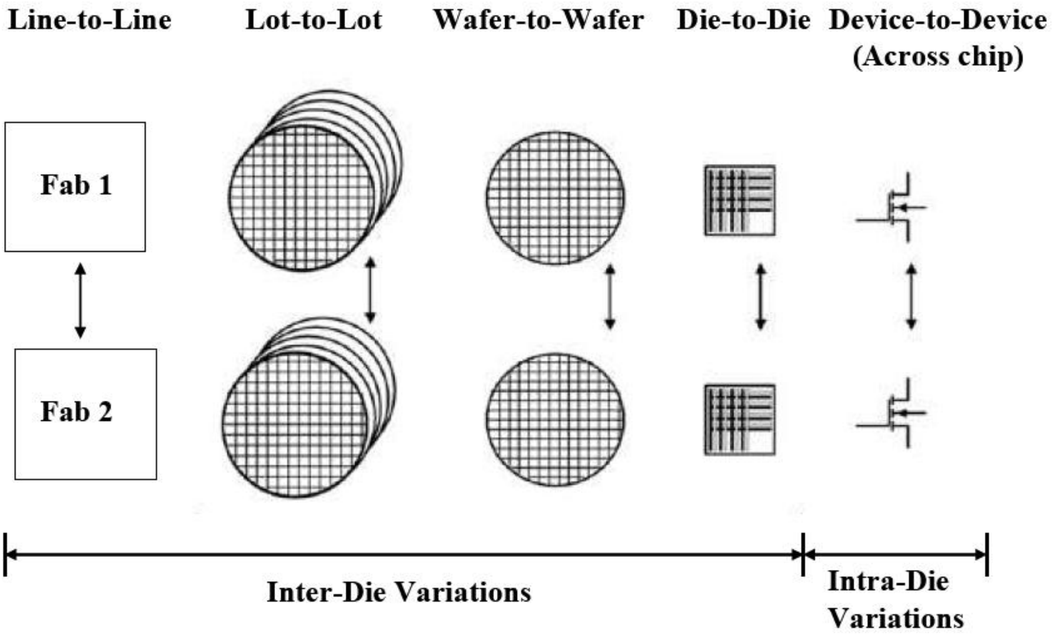


Figure 1-6 Process variability at different levels of manufacturing [23].

Some parameters shift when the equipment is loaded with a new wafer or between processing one lot of wafers to the next; on the other hand, some shift can occur between different dies in a wafer; finally the shift can also occur in between devices in a same die. Figure 1-6 shows the different spatial scales of variation as described above.

1) Inter-die variations (also referred to as die-to-die or global variation) affect the device physical parameters on the same die in a same way, and they occur from lot-to-lot, wafer-to-wafer and die-to-die. All the transistors in a given circuit are influenced uniformly by the inter-die variations, e.g., the effective channel length (L_{eff}) of all the transistors in a single die will shift in the same direction (increase or decrease) due to inter-die variations. Therefore it will not cause a mismatch between different transistors in a die.

   The sources of inter-die variations include the effective gate-length and oxide thickness variations due to the fluctuation in the time of exposure during fabrication. For design analysis purposes, it is usually assumed that each inter-die contribution is caused by different and independent sources [23, 24].

Inter-die variations have been a longstanding issue for several decades and the designers have made a lot of effort to try model this kind of uncertainty in order to make their circuits robust. The typical solution is to simulate the circuit not at one design point, but a small number of “corners” [24], which are chosen to encapsulate the behaviour of the design under worst-case conditions. This technique served the designer well in the past. However, since the semiconductor technology merged into the nanometer regions, the traditional corner-based analysis approach suffers from some major limitations, and the statistical technique becomes a potential solution for analyzing process variation effects. Details will be discussed in Chapter 2.

2) Intra-die variations (also referred to as within-die or local variation) are the deviations occurring spatially within a die, which affect the different die in a different way. These variations may have a variety of sources depending on the physics of the manufacturing process [23-25], which were negligible before technology scaled down to the nanometer regime. Nowadays, since the nanometer technology has been widely used and transistor dimensions continue to shrink towards to the next node, the intra-die variations become significant and can no longer be ignored. (In some cases even larger than inter-die variations [26].)

   Intra-die variations are mainly caused by imperfections in the mask-making process and the interaction between the lithography process and the density of shapes in a given region of the layout [27]. These variations may cause the process parameters of devices in the same die to shift in different directions, e.g., L_{eff} will increase for some transistors and decrease for the others [28]. Therefore, with the existence of intra-die variations, some part of the chip may speed up when other parts may slow down.

   Intra-die variations are design independent and in most cases related to equipment properties, wafer placement, processing temperatures etc. [29]. It is obvious that intra-die variations will result in a dimensionality problem for corner-based variation analysis since every transistor in a die requires extra corners. Since it is computationally very expensive to generate all the possible corners with such a huge increase of dimensionality, the traditional statistical analysis methodology using the Monte Carlo method becomes impractical when the intra-die variations are significant. Furthermore, the deterministic approaches fail to capture the effect of intra-die variations completely [9].

1.4.3 Spatially Correlated and Independent Variations

Intra-die variations can be further categorized into two groups based on whether they are spatially correlated or not.

1) Spatially correlated variation is when the process parameter deviation changes gradually from one location in a die to the next which may be caused by many underlying fabrication process steps. Therefore, these variations tend to affect the spatially adjacent devices in a similar manner, thus the they have more similar characteristics than those which are placed far apart [6].

2) Independent variations (also referred to as random variations) are the intra-die variation component of a device which is statistically independent from all others. They occur due to the inherent unpredictable phenomena in the semiconductor fabrication process such as random dopant fluctuations (RDF) [30, 31]. A run-time variation such as the supply voltage and operating temperature can also be treated as random components. Independent variations are hard to characterize and will cause a significant mismatch of transistors in a die.

1.5 The Impact of Process Variation

In this section, a brief survey of the impact of process variations on performance parameters will be discussed. Figure 1-7 shows the relationship between processing and device parameters and their effect on circuit and system performance. The uncertainties introduced during semiconductor fabrication as well as the operating environmental noise will be propagated all the way to the performance of system, thus affecting the product yield and cost. The scaling of CMOS devices to ultra deep sub-micron (DSM) regime will aggravate the issue of variability, which has already become a major concern in evaluating the reliability of circuits [4, 5, 23, 32].

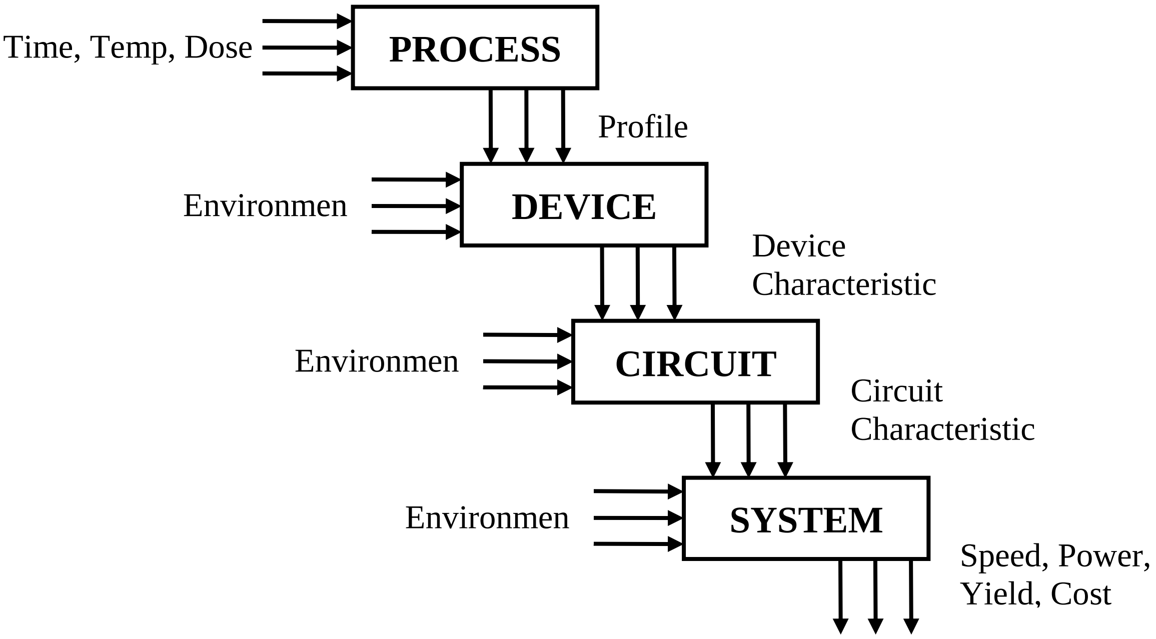


Figure 1-7 Relationship between process and device parameters and circuit and system performance.

There are a huge number of physical device parameters which can vary, it is essential to establish the components of variation that dominate each of the device and interconnect parameters. According to the literature [33, 34], the effective gate-length (L_{eff}) variation is probably the most critical device variation. The inter-die variation in gate-length is caused by the fluctuation in the duration of exposure and the intra-die variation in gate-length results from lens aberration and other lithographic effects. Both of these are significant in nanometer technology. On the other hand, device parameters such as zero-biased threshold voltage (V_{th0}), gate-oxide thickness (T_{ox}) etc, are also significant as MOSFETs are very sensitive to them. All these variations in the physical device parameters have a direct impact on the device current characteristics and threshold voltage, and subsequently the circuit characteristics such as delay and power. Figure 1-8 shows the trends in the magnitude of process variation based on the International Technology Roadmap for Semiconductors (ITRS) [35]. Virtually all technology parameters such as transistor length (L_{eff}), width (W_{eff}) and oxide thickness (T_{ox}), along with the interconnect parameters such as wire width (W), wire height (H) and resistivity (ρ) show an increasing variability over the semiconductor technology roadmap (as measured by the ratio of standard deviation over the mean value).

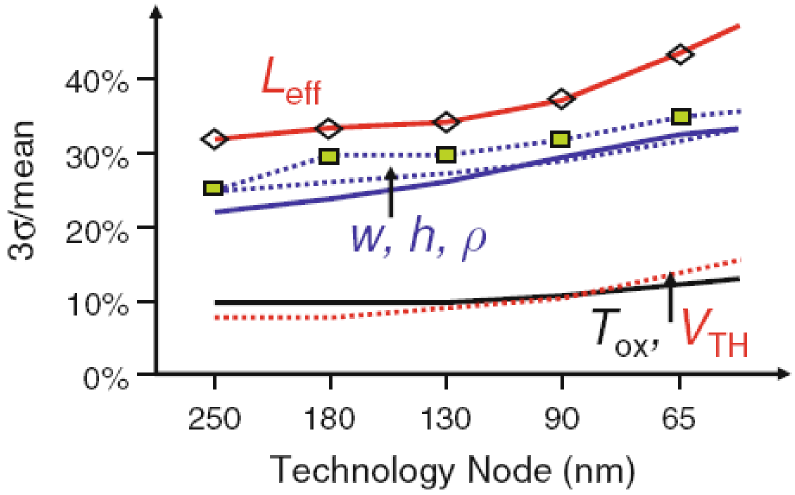


Figure 1-8 Variability trends in key process parameter with scaling process technology[35].

Although each of these parameters is important on its own, the resulting impact on the threshold voltage is what counts most from a digital circuit design perspective. As shown in Table 1-3, the threshold voltage variability is rising from 4% to 16% while evolving from 250nm to 45nm CMOS technologies. One may assume that this variation primarily results from the increasing deviations in channel length as V_{TH} is quite sensitive to variations in L. The resulting impact on both performance and power metrics is quite substantial [36].

L (nm)	*250*	*180*	*130*	*90*	*65*	*45*
V_{TH} (mV)	*450*	*400*	*330*	*300*	*280*	*200*
$\sigma(V_{TH})$ (mV)	*21*	*23*	*27*	*28*	*30*	*32*
$\sigma(V_{TH})/V_{TH}$	*4.7%*	*5.8%*	*8.2%*	*9.3%*	*10.7%*	*16%*

Table 1-3 Variation impact on device threshold voltage with scaling process technology[35].

Figure 1-9 shows the normalized distribution of the clock frequency and the leakage current of Intel microprocessors on a single wafer [37]. It can be seen that the variations in device parameters have resulted in more than a 30% frequency spread and 20x variation in the total leakage current of the chip. The highest operating frequency chips with a large leakage current and those low frequency chips with a reasonably high leakage current will have to be discarded, affecting the overall yield and cost.

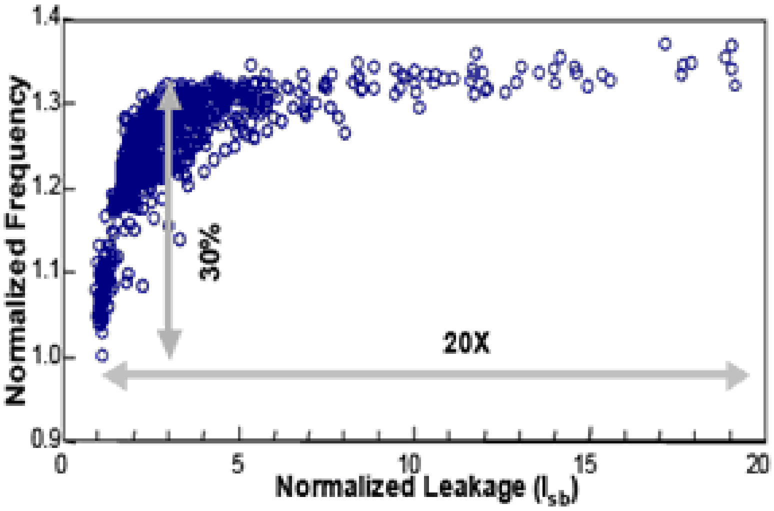


Figure 1-9 Frequency and leakage variation [37].

1.6 Motivation and Research Goals

In this chapter, an overview of process variation in semiconductor manufacturing has been given. The different sources of variations such as physical parameter variation, environmental parameter variation, model variation have been outlined. Furthermore the catalogue of different components of process variations such as inter-die and intra-die variations has also been outlined, followed by the introduction to the impact of these variations and how they propagate through the different levels of abstraction.

Process variations during manufacture will cause fluctuations in the values of the physical parameters of transistors, which is the main reason for the uncertainty in circuit delay and power dissipation. The circuit delay variation is widely recognized as the major limit to the system speed growth in today's nanometre technologies. On the other hand, the leakage power has become a significant contributor of the total circuit power consumption because of the continuous shrinking of transistor dimensions and the demand for lower power supply voltages. According to International Technology Roadmap for Semiconductors (ITRS), leakage power is expected to increase to 50% of the total chip power consumption and to dominate the switching power of a circuit over the next few technology generations. The variation of the circuit delay and leakage power consumption will significantly affect the system performance and yield. Hence there is a need to model and analyze process variation effects early in the design cycle, then modifications can be made to ameliorate these effects. With this objective in mind, the main goals and contributions of this thesis are outlined as below:

- Provide a statistical methodology to model the process variation effects at a high level of design abstraction (architectural level) in terms of propagation delay time and leakage power dissipation.
- Implement a process variability aware cell library which not only contains basic gate cells, but also more complicated functional blocks such as registers, ALUs and FIFOs in MatLab Simulink.
- Demonstrate the use of the cell library to study process variations effect for 90 nm technology on circuit delay and leakage power.
- Undertake a full timing and leakage power analysis for a 2-stage pipeline circuit using the proposed cell library, traditional statistical analysis approach and Monte Carlo simulations.
- Validate the proposed methodology through Monte Carlo simulation.
- Demonstrate the computational efficiency of the cell library compared to traditional statistical timing/power analysis and Monte Carlo technique.

It is considered that the above contributions advance the state of the art technique to analyse the effects of process variations at higher levels of abstraction.

1.7 Thesis Organization

The subsequent chapters in this thesis are organised as follows:

In Chapter 2, an overview of the analysis techniques for the effects of process variation will be outlined. The traditional worst case and Monte Carlo analysis approaches, as well as their variants, will be introduce first. The corresponding limitations of these techniques will also be discussed. Subsequently the statistical analysis methodologies will be described. The work of this thesis is based on device-to-circuit variation analysis and extends it to an architectural level, which estimates the circuit performance parameter distributions, such as delay and leakage power, due to the device parameter variations. However, the general approaches for the process-to-device variation analysis, which abstracts the variation effects during the fabrication process on the device parameters will also be briefly introduced in this chapter. The advantages and disadvantages of each analysis technique for process variation effects will be discussed.

In Chapter 3, the details of how to characterize the delay models for the standard cells, such as logic gates, will be outlined. The statistical delay model and analysis techniques will be employed. The statistical timing analysis approaches can be divided into two types: block-based and path-based techniques. The reason for using the block-based statistical timing analysis over the path-based approaches will be discussed first. Secondly, the commonly-used statistical delay models will be described and compared; thereafter details of the corresponding timing analysis methodologies using these delay models will be discussed. Subsequently, the effects of circuit operation conditions, such as input signal slope and output load capacitance, on cell delay distributions will be discussed and the corresponding cell characterization algorithm will be presented.

In Chapter 4, the leakage power characterization for the standard cells will be outlined. First, the leakage current mechanisms, which cause the unwanted leakage power dissipation when the device at a off or stand-by state, will be introduced. Followed by a comparison between the analytical and statistical leakage power models and the discussion on why the latter technique is employed. Subsequently, the statistical power analysis methodologies will be described. Several popular analysis techniques will be discussed and compared. The cell leakage power characterization algorithm will be presented at the end of this chapter.

In Chapter 5, the implementation of a statistical cell library comprising a variety of functional blocks will be outlined first. Any desired circuit can be constructed using the cell library and the process variation effects on its delay and leakage power performance can be analyzed accurately and efficiently. The methodology to characterize the higher level circuit blocks using the existing standard cells, which were introduced in Chapters 3 and 4, will be described first. The cell library implementation environment and simulation process flow will subsequently be described. Thereafter, a demonstration of using the cell library to analyze the process variation effects on the delay and leakage power performance of an example pipeline circuit will be outlined. The experimental results and the corresponding discussion will be shown in the end of this chapter.

Finally, Chapter 6 concludes the thesis with a summary of the results and possible directions for future work in this area.

1.8 Reference

[1] G. E. Moore, "Cramming more components onto integrated circuits, Reprinted from Electronics, volume 38, number 8, April 19, 1965, pp.114 ff," *Solid-State Circuits Newsletter, IEEE,* vol. 20, pp. 33-35, 2006.

[2] M. Kanellos, "New Life for Moore's Law," in *CNet News.com*, ed, April 2005.

[3] Wgsimon, "Microprocessor Transistor Counts 1971-2011 & Moore's Law," in *Wikimedia Commons*, T. C. a. M. s. L.-. 2011.svg, Ed., ed, 13 May 2011.

[4] (2007). *International Technology Road Map for Semiconductors (ITRS)*. Available: http://public.itrs.net

[5] S.-M. Kang and Y. Leblebici, *CMOS Digital Integrated Circuits: Analysis and Design*, 2nd ed.: McGraw Hill Higher Education, 1999.

[6] D. Blaauw*, et al.*, "Statistical Timing Analysis: From Basic Principles to State of the Art," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* vol. 27, pp. 589-607, 2008.

[7] B. H. Calhoun*, et al.*, "Digital Circuit Design Challenges and Opportunities in the Era of Nanoscale CMOS," *Proceedings of the IEEE,* vol. 96, pp. 343-365, 2008.

[8] D. Sylvester*, et al.*, "Invited paper: Variability in nanometer CMOS: Impact, analysis, and minimization," *Integr. VLSI J.,* vol. 41, pp. 319-339, 2008.

[9] D. S. Ashish Srivastava, David Blaauw *Statistical Analysis and Optimization for VLSI: Timing and Power*, 1st ed.: Springer, December 8, 2010.

[10] W. S. Jeroen A. Croon, Herman E. Maes, *Matching Properties of Deep Sub-Micron MOS Transistors*, 1st ed.: Springer, March 24, 2005.

[11] Z. Songqing*, et al.*, "A Probabilistic Framework to Estimate Full-Chip Subthreshold Leakage Power Distribution Considering Within-Die and Die-to-Die P-T-V Variations," in *Low Power Electronics and Design, 2004. ISLPED '04. Proceedings of the 2004 International Symposium on*, 2004, pp. 156-161.

[12] M. Alioto*, et al.*, "Analysis of the impact of process variations on static logic circuits versus fan-in," in *Electronics, Circuits and Systems, 2008. ICECS 2008. 15th IEEE International Conference on*, 2008, pp. 137-140.

[13] E. M. Conwell, *High Field Transport in Semiconductors* Academic Press Inc, October 1967.

[14] D. K. Schroder and J. A. Babcock, "Negative bias temperature instability: Road to cross in deep submicron silicon semiconductor manufacturing," *Journal of Applied Physics,* vol. 94, pp. 1-18, 2003.

[15] J. R. Black, "Electromigration: A brief survey and some recent results," *Electron Devices, IEEE Transactions on,* vol. 16, pp. 338-347, 1969.

[16] G. Nanz and L. E. Camilletti, "Modelling of chemical-mechanical polishing: a review," *Semiconductor Manufacturing, IEEE Transactions on,* vol. 8, pp. 382-389, 1995.

[17] L. K. Scheffer, "Physical CAD changes to incorporate design for lithography and manufacturability," in *Design Automation Conference, 2004. Proceedings of the ASP-DAC 2004. Asia and South Pacific*, 2004, pp. 768-773.

[18] C. A. Mack, "Understanding focus effects in submicrometer optical lithography: a review," *Optical Engineering,* vol. 32, pp. 2350-2362, Oct. 1993.

[19] Y. Jie*, et al.*, "Advanced timing analysis based on post-OPC extraction of critical dimensions," in *Design Automation Conference, 2005. Proceedings. 42nd*, 2005, pp. 359-364.

[20] P. Gupta and H. Fook-Luen, "Toward a systematic-variation aware timing methodology," in *Design Automation Conference, 2004. Proceedings. 41st*, 2004, pp. 321-326.

[21] U. D. Torben Heins, Roman Liebe and Jan Richter,, "Line edge roughness on photo lithographic masks," presented at the Metrology, Inspection, and Process Control for Microlithography, San Jose, CA, USA, 2006.

[22] R. Rao*, et al.*, "Study of Random Dopant Fluctuation Effects in FD-SOI MOSFET Using Analytical Threshold Voltage Model," *Device and Materials Reliability, IEEE Transactions on,* vol. 10, pp. 247-253, 2010.

[23] A. A. Mutlu and M. Rahman, "Statistical methods for the estimation of process variation effects on circuit operation," *Electronics Packaging Manufacturing, IEEE Transactions on,* vol. 28, pp. 364-375, 2005.

[24] S. S. Sapatnekar, "Variability and Statistical Design," *IPSJ Transactions on System LSI Design Methodology,* vol. 1, pp. 18-32, 2008.

[25] M. Eisele*, et al.*, "Intra-die device parameter variations and their impact on digital CMOS gates at low supply voltages," in *Electron Devices Meeting, 1995., International*, 1995, pp. 67-70.

[26] S. R. Nassif, "Design for variability in DSM technologies [deep submicron technologies]," in *Quality Electronic Design, 2000. ISQED 2000. Proceedings. IEEE 2000 First International Symposium on*, 2000, pp. 451-454.

[27] S. R. Nassif, "Modelling and forecasting of manufacturing variations," in *Statistical Metrology, 2000 5th International Workshop on*, 2000, pp. 2-10.

[28] S. Mukhopadhyay*, et al.*, "Modelling of failure probability and statistical design of SRAM array for yield enhancement in nanoscaled CMOS," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* vol. 24, pp. 1859-1880, 2005.

[29] K. A. Bowman*, et al.*, "Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration," *Solid-State Circuits, IEEE Journal of,* vol. 37, pp. 183-190, 2002.

[30] P. A. Stolk and D. B. M. Klaassen, "The effect of statistical dopant fluctuations on MOS device performance," in *Electron Devices Meeting, 1996. IEDM '96., International*, 1996, pp. 627-630.

[31] T. Mizuno, *et al.*, "Experimental study of threshold voltage fluctuation due to statistical variation of channel dopant number in MOSFET's," *Electron Devices, IEEE Transactions on*, vol. 41, pp. 2216-2221, 1994.

[32] B. Wong, *et al.*, *Nano-CMOS Circuit and Physical Design*, 1st ed.: Wiley-Blackwell, 2005.

[33] S. B. Samaan, "The impact of device parameter variations on the frequency and performance of VLSI chips," in *Computer Aided Design, 2004. ICCAD-2004. IEEE/ACM International Conference on*, 2004, pp. 343-346.

[34] P. S. Zuchowski, *et al.*, "Process and environmental variation impacts on ASIC timing," in *Computer Aided Design, 2004. ICCAD-2004. IEEE/ACM International Conference on*, 2004, pp. 336-342.

[35] S. R. Nassif, "Modelling and analysis of manufacturing variations," in *Custom Integrated Circuits, 2001, IEEE Conference on.*, 2001, pp. 223-228.

[36] J. Rabaey, *Low Power Design Essentials (Integrated Circuits and Systems)* Springer, 2009.

[37] S. Borkar, *et al.*, "Parameter variations and impact on circuits and microarchitecture," in *Design Automation Conference, 2003. Proceedings*, 2003, pp. 338-342.

CHAPTER 2

VARIABILITY MODELING AND ANALYSIS

2.1 Introduction

As outlined in Chapter 1, the variations which occur in the semiconductor manufacturing process of an IC can have a significant effect on its performance. Consequently it is essential to establish what the potential impact of the variations will be and so determine of the circuits will meet the specification requirements before the circuits are manufactured and tested. If the prediction indicates that the design specifications will not be satisfied it may be necessary to redesign parts of the circuit use of process with tighter tolerances or relax specifications.

With the increase in complexity of not only the manufacturing process but also the circuit designs it is essential to develop tools and techniques which enable the potential effects of process variations to be propagated and efficiently analysed throughout the design hierarchy. In this chapter process variability modelling and analysis techniques will be outlined and discussed. First of all, in section 2.2 the traditional deterministic variability-aware analysis methodologies, such as worst case and corner analysis, will be introduced; followed by a review and comparison of the Monte Carlo sampling technique and its variant approaches in section 2.3. Subsequently, several statistical techniques which have played a significant role in analyzing process variation effects will be described in sections 2.4 and 2.5, including sensitivity analysis, design of experiment (DoE) and response surface modelling (RSM). Section 2.6 outlines the process-to-device variability modelling flow based on DoE and RSM; the rest of the chapter will focus on process-to-device variability modelling methodologies such as statistical static timing and power analysis.   A brief introduction to the SPICE circuit level simulator and compact models, which are the essential tools in analyzing process variation effects in higher levels of abstraction. The chapter ends with some concluding comments.

2.2 Worst Case and Corner Analysis

It is well known that the inherent fluctuations during the integrated circuit (IC) manufacturing process results in the variation of the electrical performance of ICs. In order to make sure the circuits will behave within the design specification, it is necessary to evaluate their potential performance before fabrication. However this task is prohibitively costly in reality. The traditional solution to predict the performance of an integrated circuit is to model the process variation effects under extreme or worst case conditions which are called corners, and assuming that the IC which functions and performs satisfactorily at these extreme cases should perform properly at normal or nominal conditions [1-3]. The process of determining these worst-case conditions, and the corresponding worst-case performance, is called worst case analysis (WCA) [2].

Typically there are 4 corners for MOS transistors, FF, FS, SF and SS; “F” standing for fast indicating the best case and “S” for slow representing the worst case. Usually the first letter of the corner is associated with N-type transistors and the second letter is for P-type transistors. For example, “FS” means the all the N-type devices in the circuit are working at the best condition and all the P-type devices are working at the worst conditions.

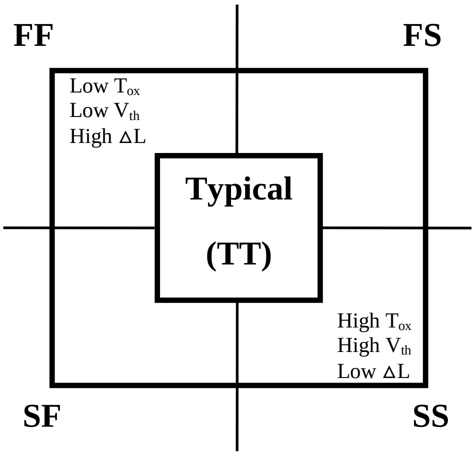


Figure 2-1 Worst case device model parameter.

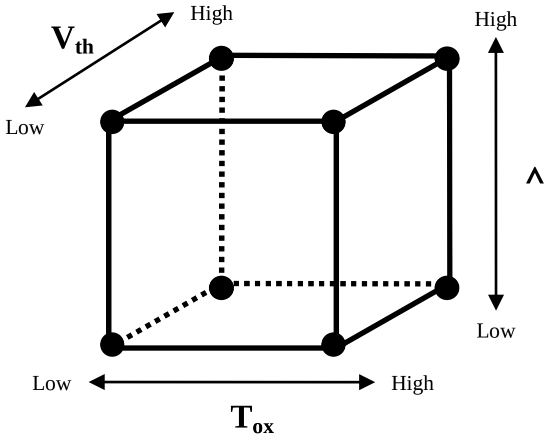


Figure 2-2 All possible corners of a transistor.

As shown in Figure 2-1, 5 different bins are defined by the 2-letter acronyms describing the relative performance characteristics of N- and P-type devices generated in accordance with the maximum and minimum values of the saturation currents and threshold voltages of the transistor [4]. When taking the correlation among process parameters into account, it is necessary to consider all the possible corners during circuit analysis since it is difficult to say which corner is the best or worst case in circuit operation. Figure 2-2 shows a 3-dimensional view of all the possible 8 corners of a device when 3 variation sources, V_{th}, T_{ox} and $\varDelta L$ are considered.

However, the traditional corner based WCA can no longer satisfy the demand of analyzing IC performance under process variations since it has several limitations. First of all, since the impact of process variations has grown, the number of the critical variation sources such as process parameters which influence the circuit behaviour has significantly increased; furthermore, the environmental variational factors such as circuit operating temperature and supply voltages have also become large contributors to the uncertainty of system performance. Therefore the total number of parameters used in WCA is large. According to [5], there are, at present, approximately 5 to 10 process parameters under variation for each type of transistor. Consequently, too many corners need to be handled when applying WCA to evaluate the circuit performance. Furthermore, the number of corners grows exponentially with the increase in the number of process parameters considered, making the corner-based WCA very computationally expensive in verifying present day nanometre technology circuits.

Secondly, WCA assumes that all the devices in a circuit work at the best and worst conditions at the same time. However, this case is extremely rare in circuit operation. Consequently the result of a WCA has a significant tendency to over or underestimate the impact of process variations on the design. Underestimation may lead to manufacturability problems and eventual loss in yield. On the contrary, overestimation makes it harder for circuits to meet their design specification leading to an increased design effort.

Finally, WCA is limited in its ability to provide designers with quantitative information about the robustness and sensitivities of their designs [6-9]. Furthermore, the corner analysis method cannot easily handle intra-die variations. All these critical limitations have resulted in significant interest in statistical modelling techniques that can be used to enable statistical analysis and performance optimization to be performed. These techniques will be discussed in the following sections.

2.3 Monte Carlo Techniques

Monte Carlo (MC) methods are a class of statistical computation algorithms that rely on repeated random sampling to compute their results. The MC method is especially useful for simulating systems with many coupled degrees of freedom, which make it the most straight forward approach [10] for characterizing random process variations, and hence finds extensive application in areas such as yield estimation. For a given function $y=f(x)$, where x is the variation source and y is the performance factor, the distribution of y due to the variation in x can readily be computed using MC analysis.

When performing an MC analysis, it is assumed that values of x can vary within the interval $[x_L, x_U]$ where x_L and x_U are the lower and upper bounds respectively. MC sampling selects a random value of x that lies in the interval. The outputs are computed for each set of input samples over hundreds or thousands of trials and the distribution for y is generated. The MC sampling approach can be readily extended to an n-dimensional design space $[x_L, x_U]^n$ in which the sample site is an ordered n-tuple. Figure 2-3 shows MC samples in a two-dimensional design space for the interval $[0, 1]^2$.

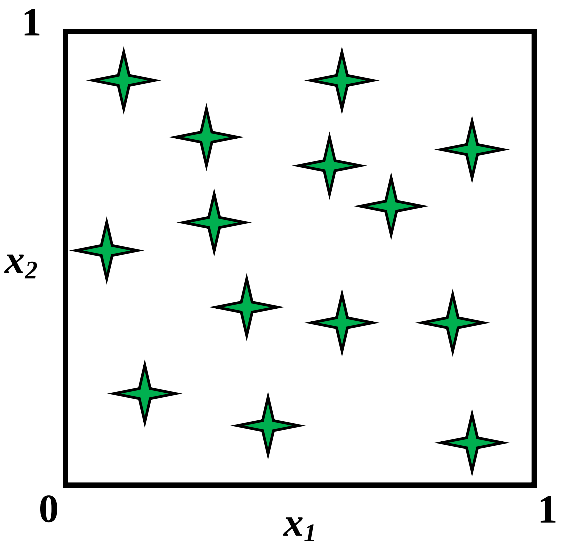


Figure 2-3 An example of Monte Carlo sampling in a two-dimensional design space for variables x_1 and x_2.

Obviously, the accuracy of the MC analysis is highly dependent on the number of sampling trials. If the sample space is large enough, it can almost cover all possible combination cases of variable values and gives a highly accurate result. However, the simplicity and accuracy of the MC technique is compromised by its expensive computational cost. Hence the computational inefficiency of MC analysis may be acceptable for small but not for larger circuits.

Due to the random and independent nature of the sample sites produced by a random number generator, sometimes a set of MC samples can often leave large regions of the design space unexplored. In order to address this drawback and improve the computational efficiency of the MC technique, several modern MC variant methods have been developed, some of which are described in the following subsections.

2.3.1 Stratified Monte Carlo Sampling

The stratified Monte Carlo sampling method was developed in an effort to provide a more uniform sampling of the design space as compared to the basic MC sampling approach [11, 12]. In the stratified MC approach, each of the *n* intervals of the design space $[x_L, x_U]^n$ has been divided into subintervals or "bins" of equal probability. All the design variables are uniformly distributed and all the bins are of equal size. After defining all the bins, a sample site then is randomly selected within each bin.

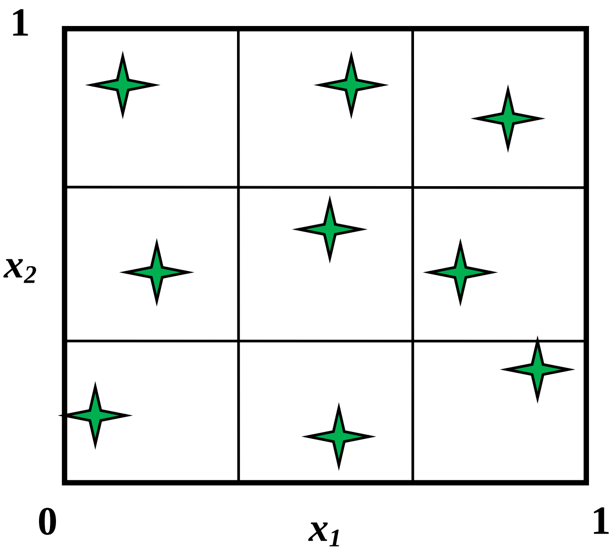


Figure 2-4 Stratified MC sampling with bin sizes having uniform probability and a sample placed randomly in each bin.

An example of using stratified MC technique is shown in Figure 2-4, where there are 2 uniformly distributed variables x_1 and x_2. The interval along each of the variables has been subdivided into 3 bins with equal size. Therefore, there are 9 bins in the interval $[0, 1]^2$. The advantage of this method is that it provides a better overall coverage of the design space compared with the basic MC analysis. Additionally it also gives flexibility in choosing the number of subintervals along each variable, which controls the number of bins in the design space. This allows the user to adjust their sampling strategy matching the available computational budget.

2.3.2 Latin Hypercube Sampling

Latin hypercube sampling (LHS) is a popular sampling strategy and another option to the MC sampling method which was first described by McKay [12] in 1979. In the context of statistical sampling, a square grid containing sample positions is a Latin square if (and only if) there is only one sample in each row and column. A Latin hypercube is the generalisation of this concept to an arbitrary number of dimensions, whereby each sample is the only one in each of the axis-aligned hyperplanes containing it.

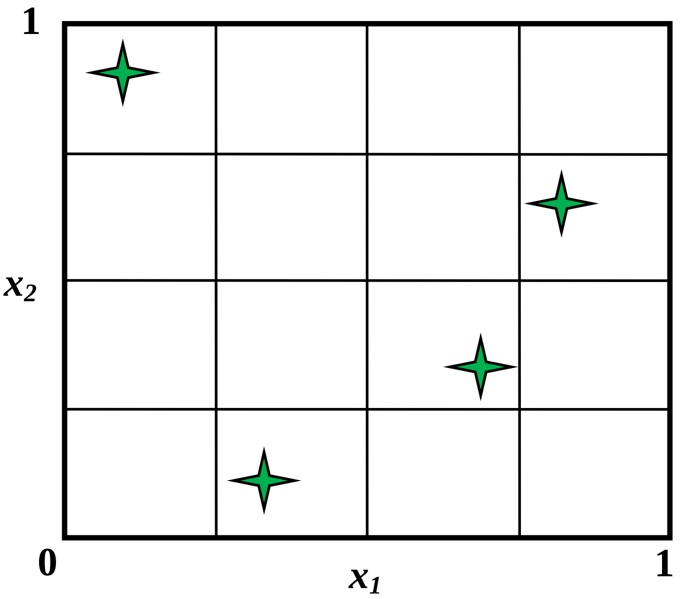


Figure 2-5 Latin Hypercube Sampling with four bins for each of the variable x_1 and x_2 .

In LHS, when sampling a function of *n* variables, the range of each variable has been divided into *m* equal intervals. Random samples are then selected in the design space to satisfy the requirements of the Latin hypercube, which means that the number of divisions, *m*, is the same for each variable. Figure 2-5 demonstrates the LHS method applied to a two-dimensional design space for variables x_1 and x_2, which are all uniformly distributed in the interval [0, 1]. As shown in the figure, *m* is equal to 4, which means there are 4 partitions in x_1 and x_2, giving a total of 16 bins. When applying LHS to this design space, the 4 samples are randomly selected with the following two conditions:

(1) Each sample is randomly placed inside a bin

(2) For all one-dimensional projections of the *m* samples and bins, there will be one and only one sample in each bin.

LHS allows the users to decide on the number of samples to match the available computational budget. The number of sample points, *m*, must be determined before sampling. On the other hand, LHS does not require more samples for more dimensions (variables). This independent characteristic is one of the main advantages of LHS. An LHS design can be built with any number of samples and not restricted to sample size that are specific multiples or powers of *n*. This computational efficiency makes LHS usable for many input variables [11].

The drawback to LHS is that there is more than one possible way to place samples in the bins in the design space to satisfy the conditions of becoming a Latin hypercube. For example, the 4 samples in Figure 2-5 can be placed in the 4 bins along either of the 2 diagonals, which leads to a nearly co-linear sample site (not random sampling anymore). In statistical jargon, this is known as highly spatial correlation. Consequently, the resulting distribution of LHS may not reflect the real characteristics of the performance parameters.

2.3.3 Quasi-Monte Carlo Sampling

The Quasi-Monte Carlo (QMS) sampling technique has recently become popular within the area of modelling variability in nanoscaled integrated circuits. In numerical analysis, the QMS approach is a method for numerical integration that is based on low discrepancy sequences (also called quasi-random or sub-random sequence). This is in contrast to the regular MC methods, which are based on sequences of pseudorandom numbers. The prefix “Quasi” refers to a sampling approach to generate sample sites in an n-dimensional space. Consequently, the selected points placed in the sampling space are as close as possible to a uniform sampling [11]. Figure 2-6 shows an example which compares the normal MC and QMC sampling in a two dimensional sampling space in the interval $[0, 1]^2$.

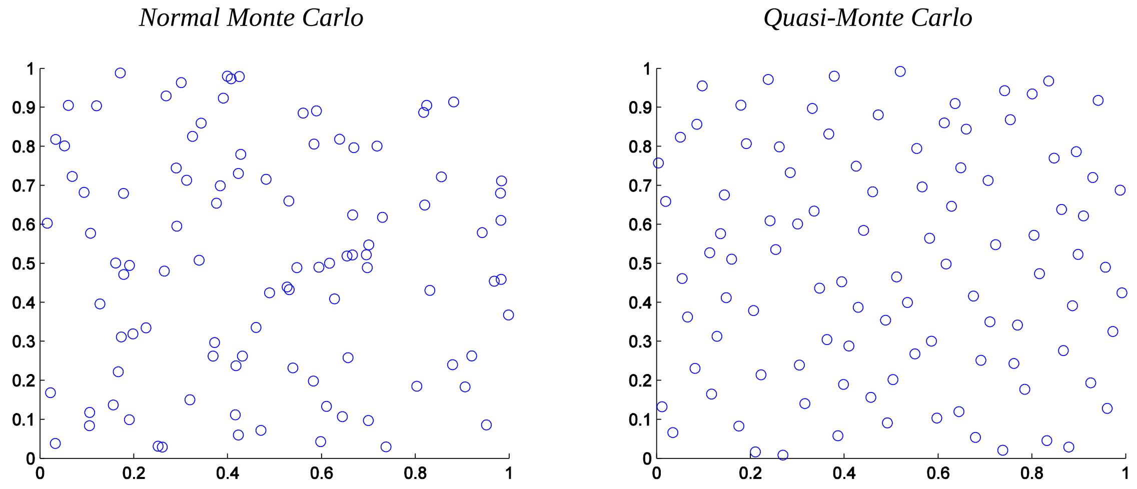


Figure 2-6 100 points from normal MC and Quasi-MC sampling.

Although a number of MC approaches or their variants (as described in this section) have been applied to analyzing the impact of process variability on circuit performance, this class of numerical analysis is still computationally very expensive regards to the massive size of nanoscaled integrated circuit and the increasing number of variation sources. In order to maintain a good coverage of such a large multi-dimensional design space, a huge number of sampling trails is required which could take hours, days or even weeks to run on a very large scaled integrated circuit. New statistical analysis techniques are desperately needed in the area of evaluating large electronic system performance. Typically the MC method plays a role as the reference for other process variation analysis techniques for validation purposes, and some of these methods are described in subsequent sections.

2.4 Sensitivity Analysis

Sensitivity analysis (SA) is simply the study of how the uncertainty in the output of a model can be apportioned to different sources of uncertainty in the model inputs [13]. Generally speaking, the SA technique investigates the robustness of a study when the study includes some form of statistical modelling. Therefore, SA can be very useful in evaluating the uncertainty of circuit performance from multiple process variation sources in a statistical manner.

For a given variation source *x* and a performance parameter *y*, the variability of *x* has been transmitted or propagated to *y* by an analytical function *f*, where *y=f(x)*. For any change in the input parameter *x*, there will be a corresponding variation in the output parameter y as shown in Equation 2.1.

$$y + \triangle y = f(x + \triangle x) \tag{2.1}$$

Assuming *x* and *y* are linearly associated, then the sensitivity of *x* with respect to *y* is the 1st order derivative of the function *f(x)*, and the standard deviation of *x* will be propagated to *y* as shown in Equation 2.2. $\triangle x$ and $\triangle y$ are the standard deviations of the parameters *x* and *y*. Equations 2.3 shows the variance propagation from *x* to *y* in the same manner.

$$\triangle y \approx \left|\frac{\partial f}{\partial x}\right| \triangle x \tag{2.2}$$

$$\sigma_y^2 \approx \left(\frac{\partial f}{\partial x}\right)^2 \sigma_x^2 \tag{2.3}$$

The numerical method used to compute the 1st derivative of the response function in sensitivity analysis is the finite difference technique. The finite difference approach is a mathematical method for approximating the solutions to differential equations using finite difference equations to approximate derivatives. For a given linear expression of the form *y=f(x)*, if the interested interval of *x* is [-Δx, Δx] then the derivatives of *f(x)* in this interval can be obtained by taking the difference quotient of the 2 sampled values of *y* at

x=-$\triangle x$ and x=$\triangle x$. The mathematical expression of this algorithm is shown in Equation 2.4, assuming *y=f(x),*

$$\frac{\partial y}{\partial x} = \frac{\partial f(x)}{\partial x} = \frac{f(x - \triangle x) - f(x + \triangle x)}{2 \triangle x} \tag{2.4}$$

The sensitivity analysis can be extended to model non-linear or multiple-order effects between the input and output parameters. However, it will lose its computational superiority due to the increase in complexity in solving the necessary high order differential equations. Additionally, the number of required samples for the numerical solution of an SA grows exponentially with the increase in the order of the response function *f(x),* which will further lead to an unacceptable computational cost. Therefore, the sensitivity-based approaches are always preferred to solve low-order relationships of parameters for most applications.

2.5 Design of Experiments and Response Surface Modelling

Design of Experiment (DoE) technique and Response Surface Methodology (RSM) [14-16] are well-established branches of statistics and have been successfully adopted since the 1920s in many manufacturing fields [14, 15]. In these techniques, a systematic method for experiment planning is used to conduct the experiments in an efficient way and enable designers to construct empirical models from which the output responses can be determined as a function of the input factors or parameters.

2.5.1 Design of Experiments (DoE)

DoE is widely used in multidisciplinary design for quality and product enhancement [14, 17], which is a procedure for choosing a set of samples in the design space, with the general goal of maximizing the amount of information gained from a limited number of samples. Statistical DoE helps to build approximations or models which yield an insight into the functional relationship between the input parameters and the performance responses of interest. The greatest advantage of DoE over MC approaches is its computational efficiency. DoE only requires a handful of experiments or simulation runs to investigate the effects of variability, which is easier and hence more feasible to generate a model for performance prediction. A designed experiment is normally described by a matrix “X”, in which the rows indicate experiment runs or simulation runs, and the columns represent the particular settings of the factors or parameters for each run. Typically, each input parameter for the experiment is represented by two levels, high (+1) and low (-1). Figure 2-7 shows an example in a design view of a 2-level DoE for 3 input parameters (x, y, z) with the design matrix which is shown in Table 4.

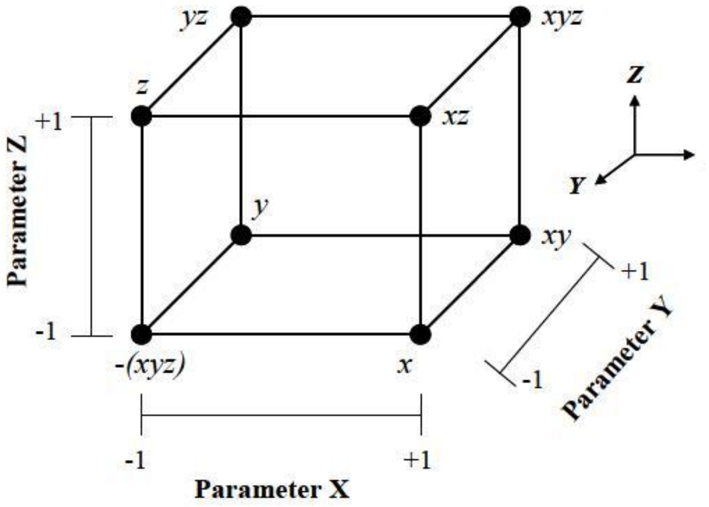


Figure 2-7 Two-level full factorial design for three factors (2^3) [18].

Table 4 Design matrix for 2^3 factorial design.

Run	Parameter			Labels
	X	Y	Z	
1	-1	-1	-1	*-(xyz)*
2	1	-1	-1	*x*
3	-1	1	-1	*y*
4	1	1	-1	*xy*
5	-1	-1	1	*z*
6	1	-1	1	*xz*
7	-1	1	1	*yz*
8	1	1	1	*xyz*

2.5.2 Response Surface Methodology (RSM)

In statistics, response surface methodology (RSM) explores the relationship between several independent variables and one or more response variables [14]. The method was introduced by G. E. P. Box and K. B. Wilson in 1951. For the response of interest, y and the vector of independent variables x included in the experimental design, influencing *y*, the relationship between *x* and *y* is described by Equation 2.5.

$$y(x) = f(x) + \varepsilon \tag{2.5}$$

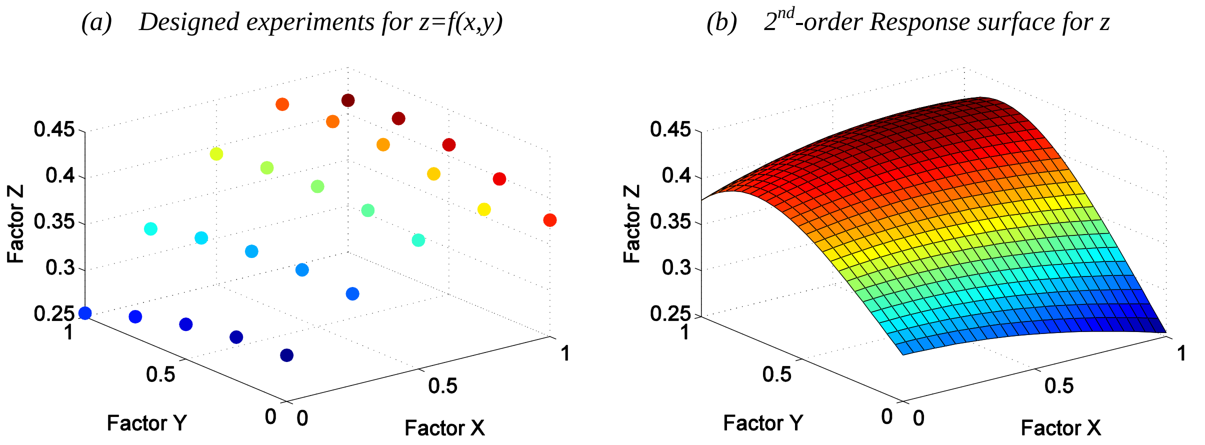
where ε represents the random error which is assumed to be normally distributed with a zero mean and unity standard deviation. The response surface function *f(x)* is approximated or predicted by a function $\hat{y}$=*g(x)*, where $\hat{y}$ is the approximation of *y*. Typically *g(x)* is expressed by a low-order polynomial. The 1st and 2nd order RSM forms are shown in Equation 2.6 and 2.7 respectively.

$$\hat{y} = g(x) = \beta_0 + \sum_{i=1}^{k} \beta_i x_i \tag{2.6}$$

$$\hat{y} = g(x) = \beta_0 + \sum_{i=1}^{k} \beta_i x_i + \sum_{i=1}^{k} \beta_{ii} x_{ii}^2 + \sum_{i=1}^{k} \sum_{j=1}^{k} \beta_{ij} x_i x_j \tag{2.7}$$

where, *k* is the number of independent input variables, x_i is the *i*th input variable, and *β* is the RSM coefficient which is calculated using least squares regression analysis to fit the response approximation $\hat{y}$.

The basic RSM approach is started from a set of designed experiments. An appropriate DoE technique is selected and applied to the input variables. Figure 2-8 (a) shows a DoE example with 25 sampling points of an arbitrary response function *z=f(x,y)*, where *x* and *y* are the input variables and *z* is the response parameter.



(a) Designed experiments for z=f(x,y)

(b) 2^{nd}-order Response surface for z

Figure 2-8 DoE and RSM in variability analysis.

After obtaining the experimental results, it is sufficient to determine which independent variables have an impact on the response variable(s) of interest. Consequently, a parameter screening step is usually required to identifying the most significant process parameters which will produce the greatest fluctuation in device electrical performance. Statistical techniques such as Pareto analysis can be used for parameter screening purposes, which compares the relative magnitude of the influence of all the main input parameters on the output responses, and arranges them in order of the decreasing absolute value of the effect. Once it is recognised that only significant variables are left, an appropriate RSM technique can be applied to model the relationship between input variables and response parameter(s) using low-order polynomials. Figure 2-8 (b) shows the 2^{nd} order RSM based on the DoE sampling points in Figure 2-8 (a).

2.6 Process-to-Device variability modelling

The statistical variation analysis approaches used from process variability to device performance are commonly based on the Design of Experiment (DoE) techniques and Response Surface Methodology (RSM). In these techniques, a systematic method for experiment planning is used to conduct the experiments in an efficient way and enable designers to construct empirical models from which the output responses can be determined as a function of the input factors or parameters. Based on this concept, the uncertainty introduced by a variety of variation sources during the manufacturing process, such as the ion implantation energy and substrate doping concentration, can be modelled as variations in the physical device parameters such as the gate channel length and threshold voltage. These variables then can be subsequently used in further circuit-level analysis to evaluate system performance parameters such as delay, power and yield.

2.6.1 The role of Technology CAD (TCAD)

Technology CAD (or Technology Computer Aided Design, or TCAD) is a branch of electronic design automation (EDA) that models semiconductor fabrication and semiconductor device operation. They are commonly used to assess the performance and the yield of an IC which is the key to the success of the IC manufacturing industry in terms of cost and time [19]. The TCAD tools are used to model the process steps, such as diffusion and ion implantation, and modelling the electrical behaviour of the devices based on the fundamental physics, such as the threshold voltage and physical dimension of the devices. TCAD may also include the creation of compact models, such as the well known SPICE transistor model, which captures the electrical behaviour of devices. The details of compact modeling will be discussed in Section 2.6.2.

TCAD tools are the essential environment for applying DoE and RSM to model process variability, which plays a significant role in Design for Manufacturing (DFM), especially after the semiconductor technology scaled down into the nanometer regime. It helps the equipment, process and circuit designer to predict the possible complications arising during the process development phase. TCAD tools contain a variety of models for device design (process models), which simulate the manufacturing steps and provide a microscopic description of device “geometry” to the device simulator. The term “geometry” means not only the device dimensions, such as the length and width of the transistor gate-channel, or whether the gate is planar, but also details inside the device structure, such as doping profiles after manufacturing. Figure 2-9 shows the output from a semiconductor process simulation for a MOSFET based on the process models used in the TCAD tools. The input to the simulator is a description of the semiconductor fabrication process and the result is the final geometry and the doping profile of the device.

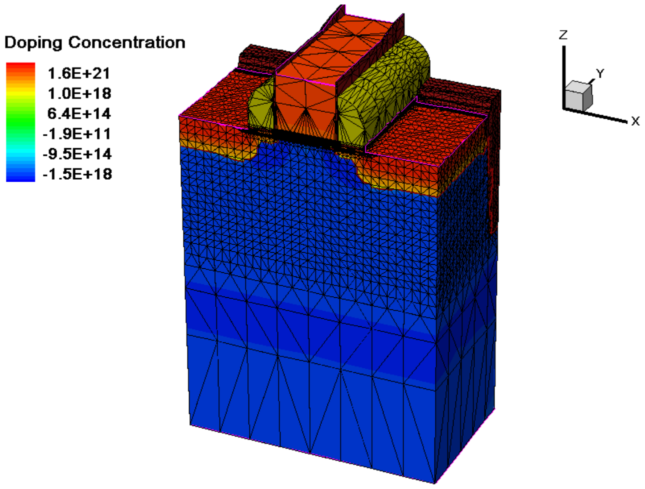


Figure 2-9 An example result from semiconductor process simulation using TCAD [20].

The use of TCAD tools starts from the physical description of integrated circuit devices, considering both physical configuration and related device performance, and then building the links between the broad range of physical and electrical behavioral models that support circuit design [20]. Physics-based modelling is an essential part of the IC process development which seeks to quantify an underlying understanding of the technology and abstract that knowledge to the device level design, such as the extraction of the key parameters that support statistical circuit performance analysis. The key advantage of TCAD is that the defined variations can simply be inserted into a computer simulation run to analyze their impact on performance. Comparatively, the experimental study of the impact of such variation is very expensive and difficult in reality.

2.6.2 Compact Model

The compact transistor model parameters are the output parameters from the process models used in the TCAD based process simulations. These models can be used by analogue circuit simulators such as SPICE to predict the electrical behaviour of a circuit being designed. The compact models include device physical parameters such as gate length and width, DC current-voltage characteristics, parasitic device capacitances, resistance and inductance, temperature effects and so on. Such models have allowed engineers to create advanced designs with first-pass success, without the need for multiple prototypes and design iterations.

The compact models for devices continuously evolve to keep up with changes in semiconductor technology. In order to standardize the model parameters used in different simulators an industry working group, called Compact Model Council (CMC) [21], was formed to maintain and promote the use of standard models. One of the famous set of compact models supported by CMC is BSIM (Berkeley Short-channel IGFET Model) series models, which have served the industry for more than 20 years. It was developed by the BSIM research group in the Department of Electrical Engineering and Computer Sciences (EECS) at the University of California, Berkeley [22-24]. BSIM3 and BSIM4 industry standard models have been widely used for the simulation of planar bulk MOSFETs. As semiconductor technology dramatically scaled, new BSIM compact models have also been developed such as BSIMSOI which used to capture the electrical characteristics of partially-depleted, fully-depleted and dynamically-depleted SOI devices.

2.6.3 TCAD-based statistical variability modelling approach

The general statistical approach based on TCAD and statistical techniques, DoE and RSM, to model the impact of process variation effects on device performance is presented in this section. The general methodology for studying variability is shown in Figure 2-10 and involves three main steps: parameter screening, model building and model analysis. The methodology begins with the calibration of the TCAD process and device electrical characteristics with the experimental data, and the extraction of the compact model parameters, such as gate channel length L and zero-biased threshold voltage V_{th0}, for given devices. This is followed by the identification of the uncontrollable process parameters which have the greatest impact on the output response being analysed, these parameters will subsequently be included in the compact model of the device.

In order to investigate the effects of process variation on a given device response, an RS model has to be created. In the RSM step, the simulation experiments are designed to thoroughly investigate and model the output responses in terms of the initially identified process parameters, or the most significant process parameters obtained from screening. Due to the fact that performing RSM analysis for a large input space requires a very large number of experimental runs (in the order of *(2^n+2n+1)*, where *n* is the number of parameters [14], it becomes computationally inefficient. In other words, screening analysis is adopted to overcome the deficiency of the RSM techniques by reducing the dimensionality of the input space. Therefore, RSM is preceded by the screening step, wherein the relatively insignificant input parameters are eliminated, since not all the input variables are influential with regard to the output response to the same degree.

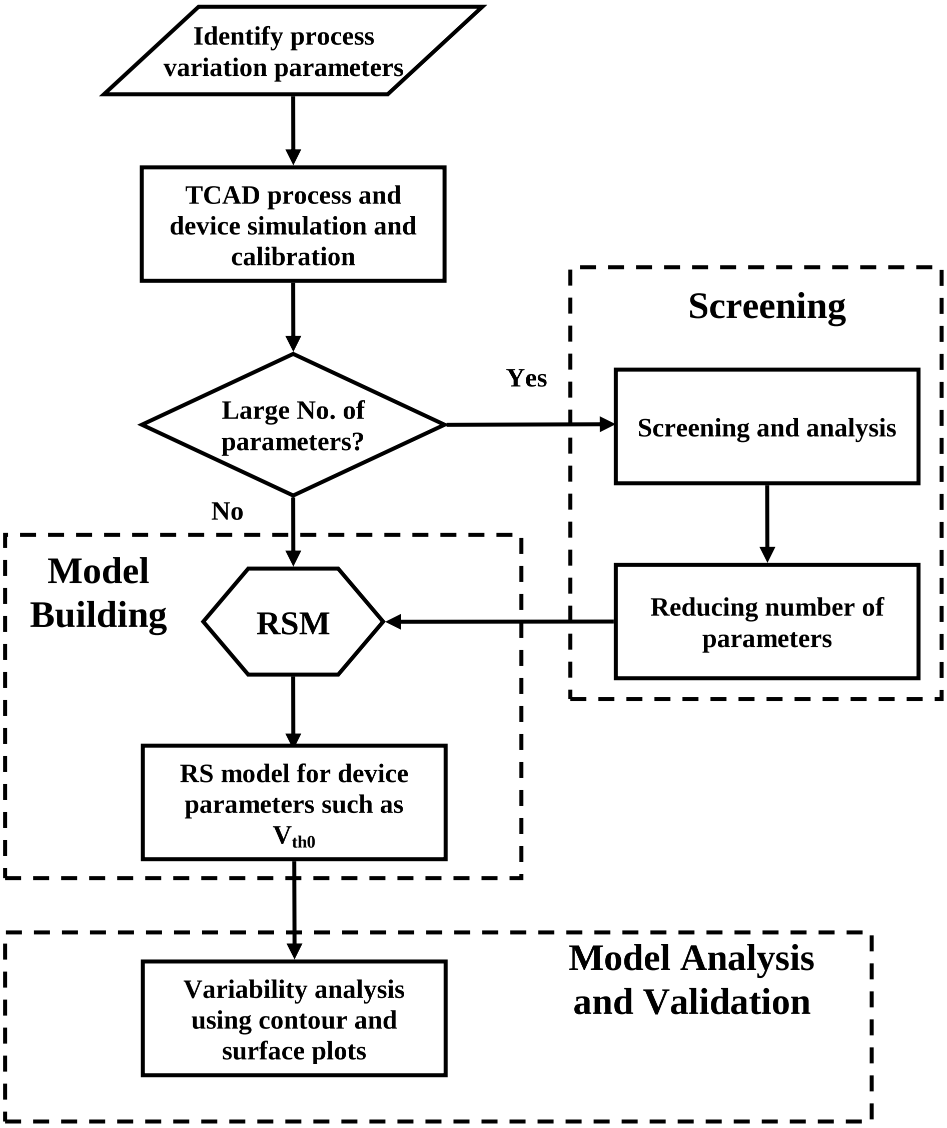

Identify process variation parameters
TCAD process and device simulation and calibration
Large No. of parameters?
Yes
No
Screening
Screening and analysis
Reducing number of parameters
Model Building
RSM
RS model for device parameters such as V_{th0}
Model Analysis and Validation
Variability analysis using contour and surface plots

Figure 2-10 Flow chart of variability analysis utilising DoE and RSM statistical techniques[18].

Finally, the RS model validity is assessed in terms of statistical residual analysis [14] such as 'goodness' of fit, which describes how well the models fit a set of observations. Measures of goodness of fit typically summarize the discrepancy between observed value and the values expected under the model in question. In the validation of the RSM model for process variability, the 2nd order fit such as such as R^2 (R-square) [14, 15] is used, where R^2 is a statistical measure between 0 to 1, which indicates how close the regression line is to the actual data points. If the value of R^2 for the RSM model is 1, which indicates a perfect estimation of the output response with no errors, the response surface plots can be generated to visualise and study the behaviour of device responses under various process variations.

In this section, a general methodology for modelling process variability from manufacturing process steps to device level parameters has been outlined. The statistical techniques of DoE and RSM are employed in a TCAD based simulation environment. The process variation effects can be visualised from the resulting response surface plots, and propagated to the device level in terms of the variation in the compact model parameters such as V_{th0}. However, present day IC designs contain a large number of transistors, it is not sufficient to analyze the impact of process variability at such a low level. The uncertainty of circuit performance parameters such as propagation delay time and leakage power dissipation must be evaluated at the early stage of the design cycle in order to prevent the possibility of significant yield loss. During the last decade, a lot of effort into statistical methodologies has been made in order to cope with the device-to-circuit variability analysis. The distributions of circuit performance parameters such as propagation delay and leakage power dissipation have been generated using statistical static timing and power analysis approaches respectively. The following sections of this chapter will give a brief introduction to these statistical techniques which tend to model the effects of process variation at a circuit level.

2.7 The role of SPICE in process variability analysis

Before introducing the device to circuit analysis approaches, variability-aware simulation tools will be discussed in this section. Most of the statistical analysis techniques for analyzing process variation effect from device level to circuit level are based on SPICE (Simulation Program with Integrated Circuit Emphasis)[25, 26]. SPICE is a general-purpose, open source analogue electronic circuit simulator developed at the Electronics Research Laboratory of the University of California, Berkeley by Laurence Nagel and Prof. Donald Pederson. It is a popular and powerful program which is widely used in integrated circuit design to determine the integrity of circuit design and to predict circuit behaviour.

With the scaling of technology down into nanometre dimensions and the increase in IC complexity, it is not practical to breadboard integrated circuits before manufacture. However, it is essential to ensure that the circuit design meets its specifications first time, as the cost of a “re-spin” is high not only in terms of the overall manufacturing costs but also lost revenues due to the delay in the product entering the market place. It should be noted, however, that with present day circuit complexities SPICE simulations of a complete design is impractical. Consequently its use is limited sub-circuits comprising several tens of transistors. When considering a complete design higher levels of simulations are invoked at either gate, Register Transfer Level (RTL) or VHDL.

SPICE, or its variants, has almost been universally accepted as a circuit analysis tool due to its versatility. It is not only capable of running DC, AC, transient, noise and sensitivity analysis in a same program, but also can adopt built-in models for diodes, bipolar transistors, JFETs and MOSFETs, including the BSIM compact models. Additionally, SPICE also provides capability to perform a worst-case sweep, Monte Carlo sweep, and automatic measurement etc., which makes it very useful and efficient in analyzing process variation effects. With the above features, difficult problems can be simulated and solved more quickly and with fewer manual errors.

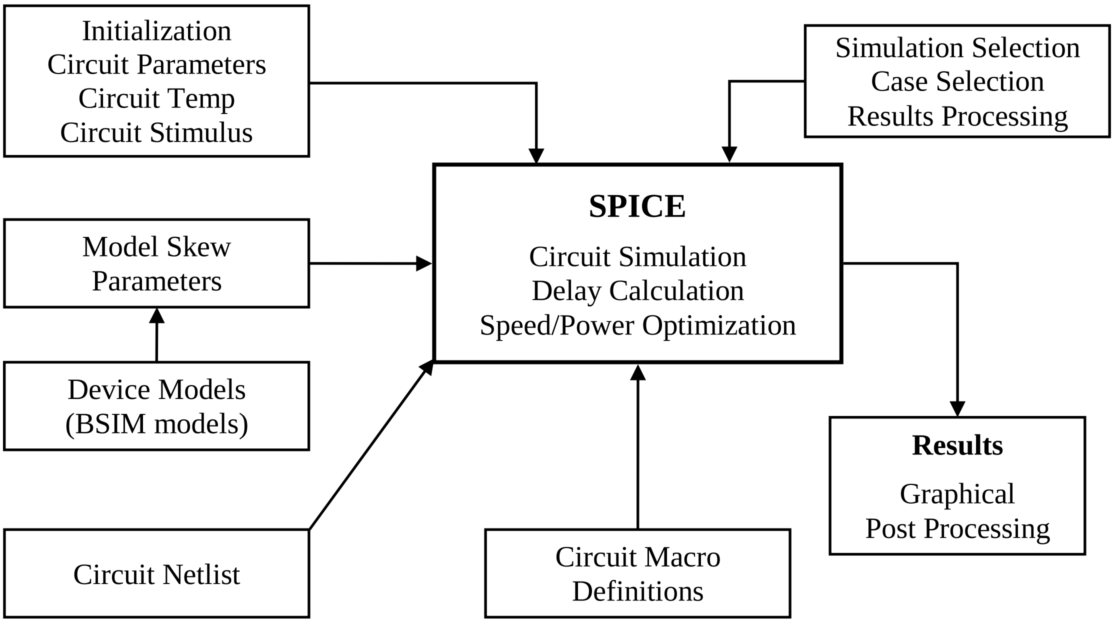


Figure 2-11 SPICE environment for integrated circuit designers.

Figure 2-11 shows the typical integrated circuit designer’s environment using the SPICE simulator. In order to run a single process - single circuit simulation, first of all, the transistor technology of the circuit needs to be selected by loading, for example, the BSIM compact models. Secondly, the circuit netlist has to be generated. Circuit macro definitions can be used in SPICE to build up the system in a hierarchical order. Subsequently, the circuit initialisation conditions need to be defined, including circuit temperature, load conditions and input stimulus. Finally, the simulation and measurement options need to be setup in order to decide the simulation type, case and how to process the result data. Nowadays the SPICE program can also support multiple processes – multiple circuits simulations. In this case, different sub-circuits in the system are treated as a local part with independent temperature, operating conditions, technology models etc.

SPICE plays a very important role in process variability analysis since it allows users to modify the values of the built-in compact model parameters and predict the corresponding circuit response in terms of delay, power and yield etc. This builds a link between the device parameter variation effects and the resulting uncertainties of the circuit characteristics. The physically measurable model parameters are called skew parameters, as shown in Figure 2-11, because they are skewed from a statistical mean to obtain the predicted performance variation. Skew parameters are generally chosen to be independent of each other so that combinations of skew parameters can be used to represent worst cases for corner analysis. The typical skew parameters for CMOS technology include transistor critical dimensions, gate oxide thickness, threshold voltage etc. On the other hand, the environmental parameter variations such as the skew of supply voltage and operating temperature are also taken into account in SPICE.

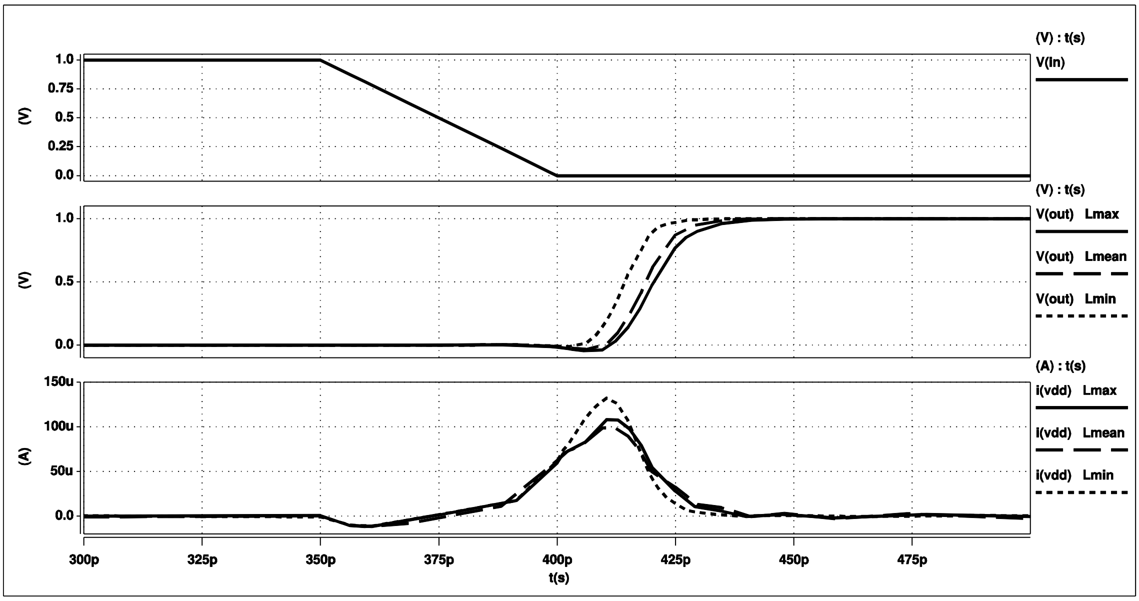


Figure 2-12 Worst case simulation using SPICE for an inverter circuit.

Figure 2-12 shows an example of the worst case analysis using SPICE when considering the variation in the gate channel length L_{eff} in an inverter circuit. The compact model used in this simulation is BSIM4.0 standard 90nm technology. The circuit has been simulated 3 times for different L_{eff} values: the minimum value (L_{min}), nominal value (L_{mean}) and maximum value (L_{max}). The variation of the inverter circuit characteristics can be observed from transient simulation results shown in Figure 2-12. The output voltage (V_{out}) waveform can indicate the delay performance and the V_{dd} current (I_{vdd}) waveform can reflect the power consumption of the inverter circuit.

The model parameter skew option in SPICE provides flexibility for the designer to analyse process variation effects using worst-case, corner and sensitivity analysis. SPICE also allows compact model parameter to be a Gaussian variable and provide multiple sampling techniques, which makes it capable of running Monte Carlo simulations to analyse the effects of device parameter variations.

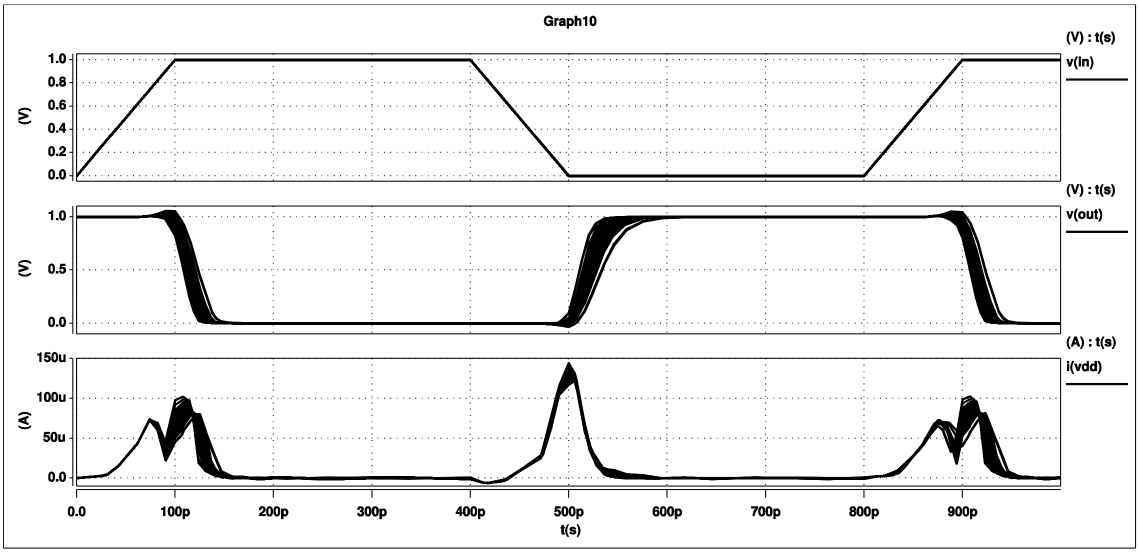


Figure 2-13 Monte Carlo simulation using SPICE for an inverter circuit.

Figure 2-13 shows the Monte Carlo simulation result of the same inverter circuit, in which 100 normally distributed random values of L_{eff} are selected. Different responses for V_{out} and I_{vdd} can be observed from the waveforms, and the measurement data can be post processed in order to plot the circuit performance PDFs. Figure 2-14 shows a delay PDF based on the experimental result in Figure 2-13. The more sample values are used, the closer the simulation results are to the actual delay distribution.

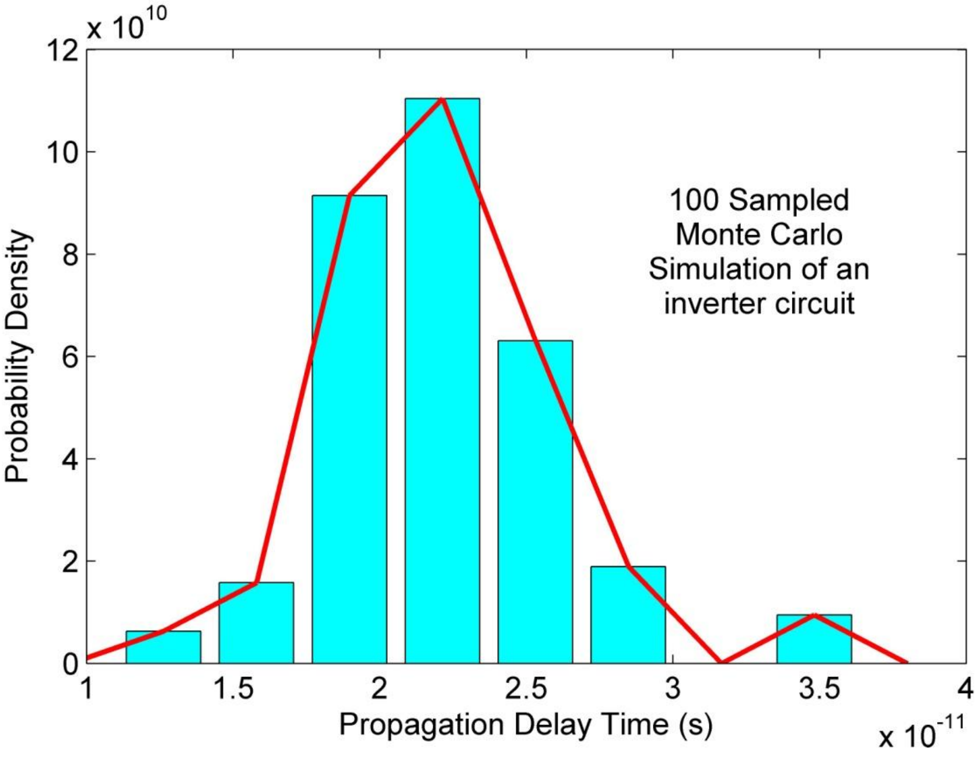


Figure 2-14 Delay PDF of an inverter circuit based on a 100 sampled MC simulation.

With the help of the SPICE program, it is possible to analyze process variation effect from transistor level to higher circuit level using a variety of approaches such as corner analysis and Monte Carlo analysis. However, as mentioned in the previous sections, these techniques are limited by their own drawbacks. During the last decade, a lot of research effort into using SPICE based statistical analysis techniques to analyze process variation effects have been made, which directly attacks the disadvantages of traditional approaches. The following 2 sections will give a brief introduction to statistical static timing and power analysis, which are the major contributions of statistical methodology to evaluate circuit reliability due to effects of process variations.

2.8 Statistical Static Timing Analysis

Since the early 1990s, static timing analysis (STA) has been widely adopted in industry to verify the speed of very-large-scale-integrated chip designs. STA is not only a universal timing sign-off tool but also plays a significant important role in numerous timing optimization techniques. STA is a deterministic approach which computes the circuit delay for a specific process condition. The fundamental weakness of STA is that, even though the global deviation in the process (inter-die variations) can be approximated using multiple corners, there is no statistical solution for modelling variations across a die (intra-die variations). Furthermore, since the semiconductor technology merges into the nanometre region, the intra-die variation has already become more and more significant and non-negligible in the total variation. In addition to the growing importance of intra-die process variations, the total number of process parameters that exhibit significant variation has also increased[27]. Consequently, even modelling of only inter-die variation in present day VLSI designs, it requires a massive number of corners[28]. Consequently, it will increase the effective runtime of STA exponentially. Finally, STA's desirable property of being conservative may be either overly pessimistic or optimistic when predicting circuit performance [29].

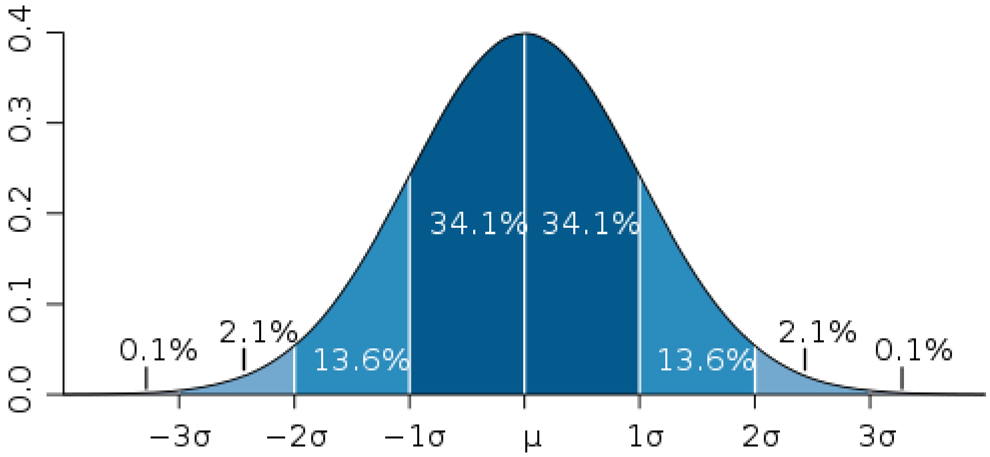


Figure 2-15 Gaussian distribution [30].

There is a need for the efficient modelling of process variations in timing analysis, which has led to extensive research in statistical STA (SSTA) during the last decade. SSTA attacks all the limitations of STA more or less directly. In SSTA, the variation sources are modelled as well known distributed variables, such as Gaussian variable for most of the cases since they are truly random. Gaussian distribution, also called the normal distribution, shown in Figure 2-15 is a continuous probability distribution that has a bell-shaped PDF, known as the Gaussian function or informally the bell curve [31]. The mathematical expression of Gaussian function is shown in Equation 2.8.

$$f(x\colon \mu,\sigma^2\,) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} \tag{2.8}$$

where μ is the mean value or nominal value of a variable x, σ is the standard deviation of x which indicates how far the variation shifts from μ. From Figure 2-15, it can be observed that about 95% of the values lie within ±2 standard deviations; and about 99.7% are within ±3 standard deviations. Hence, in practice it is assumed that all values of the Gaussian distribution are within the $\pm 3\sigma$ range, which is called 3-sigma rule [31].

SSTA uses sensitivities to find correlations among delays, and then it uses these correlations when computing how to add statistical distributions of delays. Hence, the computational complexity of SSTA grows linearly with the increase in the number of variational parameters, and intra-die variation component has been taken into account in

SSTA delay models. Additionally, a statistical delay distribution of a given circuit will be generated in SSTA rather than the worst case corners in STA. Therefore, more information is contained in SSTA results which provide more options for designers to modify their designs and balance product yields.

The initial research works for SSTA dates back to the very beginning of timing analysis in the 1960s [32] as well as the early 1990s [33, 34]. However, the majority of research work on SSTA has been done during the last 10 years with hundreds of papers published in this field since 2001. In this section, a brief review of SSTA will be outlined. It starts with the introduction to the gate delay models including first-order and higher-order forms. A description of SSTA calculation options such as "add" and "max" will follow. Finally, the classification of SSTA, path-based and block-based approaches, will be explained with the discussion of their respective advantages and disadvantages.

2.8.1 Statistical gate delay model for SSTA

In statistical gate delay modelling, the earlier solutions [44-46] are based on using discretized PDFs to handle probability distributions. However, the large number of samples in the discrete delay PDF increases the computational requirements of timing analysis and tends to degenerate into a traditional STA approach. Recently the low-order polynomial delay models [35-37, 41] have become more and more popular which can reduce a significant amount of complexity of timing analysis. In this methodology, each variation source is represented by a Random Variable (RV), which is usually distributed normally with its mean value μ and variance σ^2. These variation sources can be the physical parameters of transistors such as effective gate channel length L_{eff}, threshold voltage V_{th} etc; or the circuit environmental parameters such as operating temperature T, supply voltages V_{dd} etc. On the other hand, the response parameter with respect to the source RVs is the gate performance in terms of propagation delay. If the response parameter is a close-to-linear combination of the source RVs, then it is also assumed to be normal variable. By using sensitivity analysis, the sensitivities of the source variables with respect to the response variable can be calculated using a small number of designed experiments (SPICE simulation runs). This enables the use of a first-order polynomial (Canonical form) to represent the gate delay distributions, shown in Equation 2.9 [35]:

$$D(Delay) = \mu_D + \sum_{i=1}^{n} \beta_{Di} G_i + \beta_{D(n+1)} R \tag{2.9}$$

where μ_D is mean delay time of the gate. G_i represents the i^{th} global variational source (Inter-die); these RVs are shared by all the gates in the same die. R is the sum of all the local RVs in the gate (Intra-die); these RVs are independent among different gates so that they can be combined into one RV. β_D*'s* are the sensitivity coefficients for all the RVs in this delay model. All the RVs in a canonical gate delay model follow a normal distribution (Gaussian). Since the linear combination of normal RVs is still normal, then the gate delay modelled by Equation 2.9 is also a normal RV.

Zhang et al [36] points out that if the parameter variation is greater than 30% of its mean value ($3\sigma/\mu>30\%$), then the first order delay model will become inaccurate; consequently, a higher order quadratic delay model is required. Equation 2.10 shows the form of the quadratic gate delay model [37]:

$$D(Delay) = \mu_D + {\beta_g}^* \delta_g + {\delta_g}^* \Gamma_g \delta_g + \beta_{D(n+1)} R \tag{2.10}$$

where $\delta_g = [G_1, G_2, \ldots, G_p]^*$ is the variable vector for ‘p’ global variation sources, and “*” represents the transpose operation. The vector β_g and matrix Γ_g are only vectorized representation of the Taylor expansion coefficients as shown in Equation 2.11 [37].

$$\beta_g(i) = \frac{\partial D_g}{\partial G_i} \quad and \quad \Gamma_g(i,j) = \frac{1}{2}\frac{\partial^2 D_g}{\partial G_i \partial G_j} \tag{2.11}$$

Since the intra-die variation is independent and behaves close to linearly, the intra-die variation component in the quadratic delay form is the same as it is in the canonical form (Equation 2.9). However, the complexity of both the polynomial fitting and delay calculations using the quadratic model grows exponentially with the increase in the number of variation sources. This drawback has emphasised the major limitation of SSTA compared with STA, the high computational complexity, which is heavily criticised by industry. Therefore, most of the SSTA approaches tend to use the 1st order canonical delay form.

2.8.2 Timing graph and SSTA operations

The timing analysis procedure requires an abstraction of a timing graph from the circuit under analysis. A timing graph is a directed acyclic graph (DAG) which has no directed cycles. That is, it is formed by a number of nodes and directed edges and each edges is connected between 2 nodes, there is no way to start at some node *x* and follow a sequence of edges that eventually loops back to *x* again [38]. The nodes in the timing graph represent the gate input and output pins. The weights of the edges represent the timing parameters in the circuit, namely the gate input pin-output pin delay and wire delay between gates. Figure 2-16 shows an example of a combinational circuit and its timing graph. Typically, in a timing graph, all the primary input signals are connected to a virtual source node and all the primary output signals are connected to a virtual sink node as shown in Figure 2-16.Therefore, the resulting timing graph has a signal source and sink node for computational convenience.

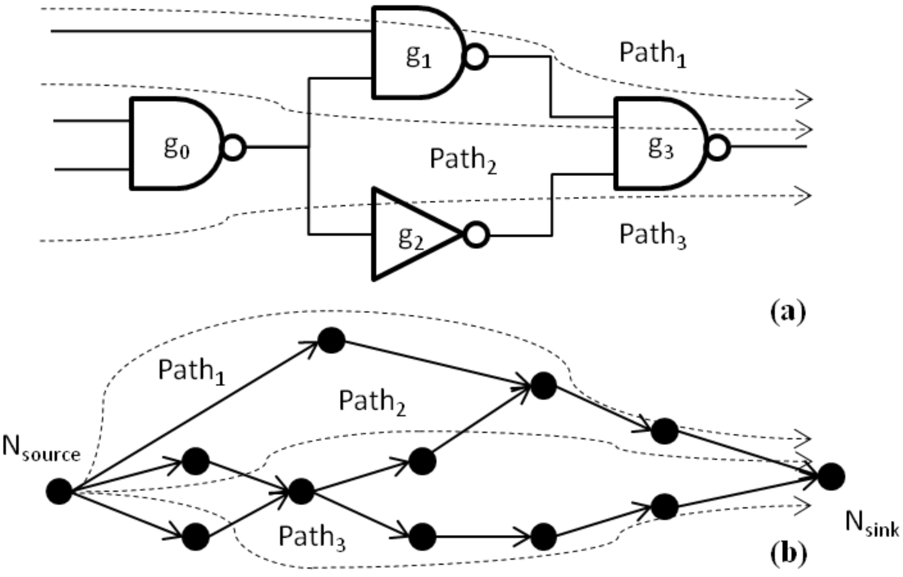


Figure 2-16 Example circuit in (a) and its timing graph in (b).

The timing graph constructed for a sequential circuit is similar. Figure 2-17 shows an example of a sequential circuit and the corresponding timing graph. The path delay has been divided into several combinational delay parts by the clock signal. All the delays including clock-to-q delay and setup times of the sequential elements are again modelled using weights on their corresponding graph edges. The virtual source node corresponds to the input driver of the on-chip clock network. The virtual sink node also corresponds to the clock input driver, and the capture path is represented by nodes with negative weighted edges in the timing graph.

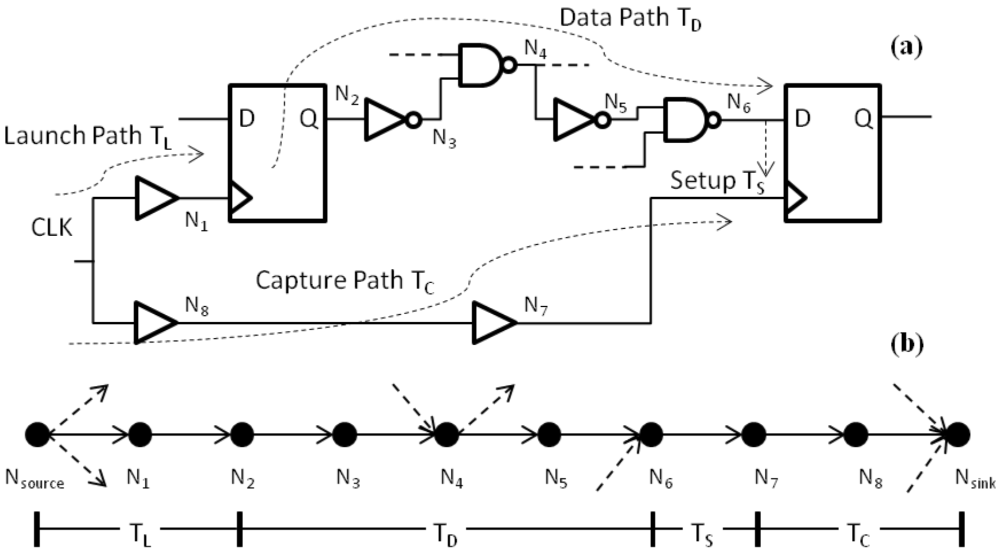


Figure 2-17 Timing elements of a sequential circuit path (a) and its timing graph (b) [29].

In SSTA, device parameters such as gate length, oxide thickness and doping concentrations are modelled as RVs. In order to extend the concept of the timing graph to a statistical abstraction, the weight of each delay edge must be treated as a function of these variational parameters. The definition of statistical timing graph is as follows:

> *"A timing graph $G = \{N, E, n_s, n_f\}$ is a directed graph having exactly one source node n_s and one sink node n_f, where N is a set of nodes, and E is a set of edges. The weight associated with an edge corresponds to either the gate delay or the interconnect delay. The timing graph is said to be statistical timing graph if i^{th} edge weight d_i is an RV [29]."*

There are 2 types of basic operations in SSTA, “Add” and “Max”. The “Add” operation is used in the summation of all the weights of the delay edges in the same signal path. If all the timing quantities are modelled as normal variables, the result of an “Add” is also a normal variable. The main difficulty with SSTA is focused on the statistical “Max” operation, which is used to compute the output delay distribution in SSTA when multiple edges converge on the same node. Since the output result of the non-linear “max” operation is no longer in the original polynomial form as the given input signal, the timing analysis cannot continue to the next node. Most of the proposed solutions are to match the first two moments of the “max” result polynomial to their analytical values, whose expressions are derived by C. E. Clark [39] in 1961as shown in Equations 2.12 - 2.16:

$$E[\max(b,c)] = \mu_b\Phi(\theta) + \mu_c\Phi(-\theta) + \theta\varphi\left[\frac{\mu_b - \mu_c}{\theta}\right] \tag{2.12}$$

$$var[\max(b,c)] = (\sigma_b^2 + \mu_b^2)\Phi(\theta) + (\sigma_c^2 + \mu_c^2)\Phi(-\theta) + (\mu_b + \mu_c)\theta\varphi\left[\frac{\mu_b-\mu_c}{\theta}\right] - \{E[\max(b,c)]\}^2 \tag{2.13}$$

Where

$$\Phi(y) \equiv \int_{-\infty}^{y} \varphi(x)dx \tag{2.14}$$

$$\varphi(x) \equiv \frac{1}{\sqrt{2\pi}}\exp\,(-\frac{x^2}{2}) \tag{2.15}$$

$$\theta \equiv (\sigma_b^2 + \sigma_c^2 - 2\rho\sigma_b\sigma_c)^{\frac{1}{2}} \tag{2.16}$$

The goal of doing this is to re-express the non-normal “max” result back to a normal form again, then keep the SSTA alive through the whole circuit under analysis. Therefore, the resulting distributions of SSTA can only be approximated.

2.8.3 Block-based and path-based SSTA

A lot of research into statistical timing analysis has been made during the last decade [35-37, 40-43]. In SSTA the variational process parameters are described as random variables (RV), such as normal Gaussian variables in most of the cases; the gate delay is usually modelled as low-order polynomials of all the RVs, which can take both inter-die (global) and intra-die (local) variations into account [40]. Basically, SSTA can be sorted into 2 classes; first path based SSTA [42] wherein propagation delay times of all possible paths in the circuit will be calculated respectively, then the slowest path can be identified. The computational complexity of the path-based approach is low because it requires only one “max” operation no matter how large the circuits are. However it is very difficult to establish all the possible paths of a circuit especially for very large ICs. Additionally, the correlation between signal paths is totally ignored in path-based SSTA. Furthermore, path based methodology does not lend itself to incremental processing so that it will lose efficiency when applied to larger circuits. The second class of SSTA is the block based approach [35-37, 41] which can propagate delay distribution from primary inputs to the primary outputs of cells or functional blocks in topological order. The block-base SSTA can lead to incremental processing making it easy to analyze the circuit in a hierarchical manner, and there is no need for path selection. The biggest disadvantage of block-based SSTA is that it requires running statistical “max” operation frequently, which is mathematically a hard technical problem normally with a high computational complexity. Most of the research work into block-based SSTA focuses on how to solve the statistical “max” operation. Recently, a simplified Monte Carlo based SSTA has been proposed [43], which could be a new solution to SSTA. These SSTA techniques will be compared and discussed in detail in Chapter 3.

2.9 Statistical Leakage Power Analysis

The static or leakage power dissipation has become a significant contributor of the total circuit power consumption because of the continuous shrinking of transistor dimensions and the demand for lower power supply voltages. According to the International

Technology Roadmap for Semiconductors (ITRS), leakage power is expected to increase to 50% of the total chip power consumption and to dominate the switching power of a circuit over the next few generations.

Similar to the timing analysis issues, the traditional corner-based analysis technique can no longer satisfy the demands of leakage power characterization in modern CMOS integrated circuits since the analysis result is either too pessimistic or optimistic, and it cannot easily handle the correlations between parameters. A number of research works into statistical power analysis (SPA) has been carried out during the last decade in order to meet the shortfalls of the corner-based approach.

SPA is used to calculate the total circuit power dissipation by taking the summation of the power consumption of every cell in the circuit. Just like SSTA, SPA also uses RVs to represent device parameter variations and the gate leakage power models are treated as low-order polynomials. However the leakage current, which is the cause of the undesired power dissipation when circuits are in a static state, has an exponential relationship with most of the sensitive device parameters. Consequently, the distribution of leakage power dissipation due to the normal distributed process variation has a lognormal form. The canonical gate leakage power model is shown in Equation 2.17 below:

$$P = exp\left(\mu + \sum_{i=1}^{n} \beta_i G_i + \beta_{(n+1)} R\right) \tag{2.17}$$

The model is very similar to the canonical gate delay form in Equation 2.9 as discussed in the previous section. μ is mean leakage power of the gate. G_i represents the i^{th} global variational source (Inter-die) and R is the sum of all the local RVs in the gate (Intra-die); β_i is the sensitivity coefficient for the corresponding RVs in this leakage power model.

By contrast, the amount of investigation into SPA is small compared to the research into SSTA since the power analysis is mathematically easier than timing analysis without nonlinear “Max” operation. The basic SPA approach is based on Wilkinson’s method [47] and its extension [48], both of these approaches provide good accuracy but with an overall complexity equal to $O(n^2)$, where n is the number of gates in the circuit. A recursive technique has been reported in [49-52], which can significantly reduce the computation time of SPA; this makes it possible to apply SPA in a large circuit. The details about SPA methodologies will be introduce in Chapter 4.

2.10 Summary

The traditional worst case and corner based analysis approaches have been reviewed. These deterministic techniques are suffering from some major limitations, as discussed in Section 2.2, for analyzing IC performance when process variation effects become more serious in nanometre technology implementations. The Monte Carlo technique could be an alternative solution for variability-aware analysis. However, MC methodology requires a significantly long computational time in order to maintain the accuracy of the analysis results for larger circuits. Under this circumstance, the statistical analysis technique becomes a better choice for evaluating the effects of process variation on circuit performance.

The process-to-device variation analysis is commonly based on the design of experiments and response surface modelling approaches. This type of analysis can extract the variability effects from the process parameters to device parameters. DoE and RSM is more efficient than the analytical approach since the most accurate models are based on simulation. Moreover, these techniques provide a reasonable balance between accuracy and the computational efficiency as compared to MC simulations.

The device-to-circuit variation analysis can predict the distributions of the circuit performance parameters such as delay and leakage power. Typically, the basic cell in this analysis is a logic gate whose performance is commonly modeled as a low-order polynomial (canonical model). The variational sources for the polynomials are the device parameters represented by Gaussian variables. The circuit delay and leakage power performance can be evaluated by SSTA and SPA based on the canonical model. Higher-order models can be applied in order to improve the accuracy but the computational complexity will increase exponentially. SSTA is used to propagate the timing variations through the timing graph and SPA is used for summing the leakage power dissipation for all the gates in a circuit. However, analyzing circuit performance at such a low level, such as gate level, is inefficient because of the massive size of the present day ICs. Therefore, higher level analysis is essential.

The work in this thesis is aiming to model process variation effects at a architectural level, the device parameter variations being assumed given. The propagation delay and leakage power dissipation are chosen to be performance parameters of system. Hence, the canonical model and SSTA/SPA introduced in this chapter are fundamental to achieving this goal.

2.11 Reference

[1] S. S. Sapatnekar, "Variability and Statistical Design," *IPSJ Transactions on System LSI Design Methodology,* vol. 1, pp. 18-32, 2008.

[2] S. R. Nassif*, et al.*, "A Methodology for Worst-Case Analysis of Integrated Circuits," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* vol. 5, pp. 104-113, 1986.

[3] A. Nardi*, et al.*, "Impact of unrealistic worst case modeling on the performance of VLSI circuits in deep submicron CMOS technologies," *Semiconductor Manufacturing, IEEE Transactions on,* vol. 12, pp. 396-402, 1999.

[4] A. A. Mutlu and M. Rahman, "Statistical methods for the estimation of process variation effects on circuit operation," *Electronics Packaging Manufacturing, IEEE Transactions on,* vol. 28, pp. 364-375, 2005.

[5] J. Watts, "Modeling circuit variability," in *Physics of Semiconductor Devices, 2007. IWPSD 2007. International Workshop on*, 2007, pp. 57-61.

[6] S. G. Duvall, "Statistical circuit modeling and optimization," in *5th International Workshop on Statistical Metrology*, 2000, pp. 56-63.

[7] K. Singhal and V. Visvanathan, "Statistical device models from worst case files and electrical test data," *Semiconductor Manufacturing, IEEE Transactions on,* vol. 12, pp. 470-484, 1999.

[8] J. A. Power*, et al.*, "Relating statistical MOSFET model parameter variabilities to IC manufacturing process fluctuations enabling realistic worst case design," *Semiconductor Manufacturing, IEEE Transactions on,* vol. 7, pp. 306-318, 1994.

[9] A. Dharchoudhury and S. M. Kang, "Worst-case analysis and optimization of VLSI circuit performances," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* vol. 14, pp. 481-492, 1995.

[10] R. VanSlyke, "Monte Carlo methods and the Pert Problem," *Operations Research,* vol. 11, pp. 839-860, 1963.

[11] A. A. Giunta*, et al.*, "Overview of Modern Design of Experiments Methods for Computational Simulations," in *41st AIAA Aerospace Sciences Meeting and Exhibit*, Reno, NV, USA, 2003.

[12] M. D. McKay*, et al.*, "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics,* vol. 21, pp. 239-245, 1979.

[13] A. Saltelli*, et al.*, *Global sensitivity analysis the primer*, 1st ed.: Wiley-Interscience, 2008.

[14] D. C. Montgomery, *Design and Analysis of Experiments*, 6th ed.: Wiley, 2004.

[15] G. E. P. Box and N. R. Draper, *Empirical Model-Building and Response Surfaces*, 1st ed.: Wiley, 1987.

[16] Y. Aoki, *et al.*, "A New Design-Centering Methodology for VLSI Device Development," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 6, pp. 452-461, 1987.

[17] S.-M. Kang and Y. Leblebici, *CMOS Digital Integrated Circuits: Analysis and Design*, 2nd ed.: McGraw Hill Higher Education, 1999.

[18] S. Shedabale, "Statistical Modelling of Process Variations in CMOS Devices and Circuits," Phd dissertation, Electrical Electronic and Computer Engineering, Newcastle Universiy, 2009.

[19] *International Technology Road Map for Semiconductors (ITRS)*. Available: http://public.itrs.net

[20] L. Lavagno, *et al.*, *Electronic Design Automation for Integrated Circuits Handbook*, 1st ed. vol. 2: CRC Press, 2006.

[21] *Compact Model Council (CMC)*. Available: http://engineering.techamerica.org/cmc-compact-model-council/

[22] B. J. Sheu, *et al.*, "BSIM: Berkeley short-channel IGFET model for MOS transistors," *Solid-State Circuits, IEEE Journal of*, vol. 22, pp. 558-566, 1987.

[23] Y. Cheng and C. Hu, *MOSFET Modeling and BSIM3 User's Guide*, 1st ed.: Springer, 1999.

[24] *BSIM (Berkeley Short-channel IGFET Model) Group*. Available: http://www-device.eecs.berkeley.edu/bsim/

[25] L. W. Nagel and D. O. Pederson, "Simulation Program with Integrated Circuit Emphasis," in *16th Midwest Symp. Circ. Theory*, Waterloo, Canada, 1973.

[26] L. W. Nagel, "SPICE2: A Computer Program to Simulate Semiconductor Circuits," PhD dissertation, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, CA, 1975.

[27] S. R. Nassif, "Modeling and forecasting of manufacturing variations," in *Statistical Metrology, 2000 5th International Workshop on*, 2000, pp. 2-10.

[28] L. Scheffer, "The count of Monte Carlo," in *TAU int. Workshop Timing*, 2004.

[29] D. Blaauw, *et al.*, "Statistical Timing Analysis: From Basic Principles to State of the Art," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 27, pp. 589-607, 2008.

[30] Mwtoews, "Normal distribution curve that illustrates standard deviations," in *Based (in concept) on figure by Jeremy Kemp in 2005*, 1st ed, 2007.

[31] G. Casella and R. L. Berger, *Statistical Inference*, 2nd ed.: Duxbury Press, 2001.

[32] T. I. Kirkpatrick and N. R. Clark, "Pert as an Aid to Logic Design," *IBM Journal of Research and Development*, vol. 10, pp. 135-141, 1966.

[33] H. F. Jyu, *et al.*, "Statistical timing analysis of combinational logic circuits," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 1, pp. 126-137, 1993.

[34] R. B. Brawhear, *et al.*, "Predicting circuit performance using circuit-level statistical timing analysis," in *European Design and Test Conference, 1994. EDAC, The European Conference on Design Automation. ETC European Test Conference. EUROASIC, The European Event in ASIC Design, Proceedings.*, 1994, pp. 332-337.

[35] C. Visweswariah, *et al.*, "First-Order Incremental Block-Based Statistical Timing Analysis," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 25, pp. 2170-2180, 2006.

[36] Z. Lizheng, *et al.*, "Correlation-preserved non-Gaussian statistical timing analysis with quadratic timing model," in *Design Automation Conference, 2005. Proceedings. 42nd*, 2005, pp. 83-88.

[37] Y. Zhan, *et al.*, "Correlation-aware statistical timing analysis with non-Gaussian delay distributions," in *Design Automation Conference, 2005. Proceedings. 42nd*, 2005, pp. 77-82.

[38] N. Christofides, *Graph Theory. An Algorithmic Approach*, illustrated ed.: Academic Press Inc, 1975.

[39] C.E.Clark, "The greatest of a finite set of random variables," *Oper.Res*, vol. 9, pp. 145-162, 1961.

[40] K. Okada, *et al.*, "A statistical gate-delay model considering intra-gate variability," in *Computer Aided Design, 2003. ICCAD-2003. International Conference on*, 2003, pp. 908-913.

[41] S. Bhardwaj, *et al.*, "A Framework for Statistical Timing Analysis using Non-Linear Delay and Slew Models," in *Computer-Aided Design, 2006. ICCAD '06. IEEE/ACM International Conference on*, 2006, pp. 225-230.

[42] C. S. Amin, *et al.*, "Statistical static timing analysis: how simple can we get?," in *Design Automation Conference, 2005. Proceedings. 42nd*, 2005, pp. 652-657.

[43] A. Singhee, *et al.*, "Practical, fast Monte Carlo statistical static timing analysis: Why and how," in *Computer-Aided Design, 2008. ICCAD 2008. IEEE/ACM International Conference on*, 2008, pp. 190-195.

[44] L. Jing-Jia, *et al.*, "Fast statistical timing analysis by probabilistic event propagation," in *Design Automation Conference, 2001. Proceedings*, 2001, pp. 661-666.

[45] L. Jing-Jia, *et al.*, "False-path-aware statistical timing analysis and efficient path selection for delay testing and timing validation," in *Design Automation Conference, 2002. Proceedings. 39th*, 2002, pp. 566-569.

[46] S. R. Naidu, "Timing yield calculation using an impulse-train approach," in *Design Automation Conference, 2002. Proceedings of ASP-DAC 2002. 7th Asia and South Pacific and the 15th International Conference on VLSI Design. Proceedings.*, 2002, pp. 219-224.

[47] S.C.Schwartz and Y.S.Yeh, "On the distribution function and moments of power sums with lognormal components," *Bell Syst. Tech. J.*, vol. 61, pp. 1441-1462, 1982.

[48] A. A. Abu-Dayya and N. C. Beaulieu, "Outage probabilities in the presence of correlated lognormal interferers," *Vehicular Technology, IEEE Transactions on*, vol. 43, pp. 164-173, 1994.

[49] A. Srivastava, *et al.*, "Statistical optimization of leakage power considering process variations using dual-Vth and sizing," in *Design Automation Conference, 2004. Proceedings. 41st*, 2004, pp. 773-778.

[50] R. Rao, *et al.*, "Statistical estimation of leakage current considering inter- and intra-die process variation," in *Low Power Electronics and Design, 2003. ISLPED '03. Proceedings of the 2003 International Symposium on*, 2003, pp. 84-89.

[51] A. Srivastava, *et al.*, "A Novel Approach to Perform Gate-Level Yield Analysis and Optimization Considering Correlated Variations in Power and Performance," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 27, pp. 272-285, 2008.

[52] C. Hongliang and S. S. Sapatnekar, "Full-chip analysis of leakage power under process variations, including spatial correlations," in *Design Automation Conference, 2005. Proceedings. 42nd*, 2005, pp. 523-528.

CHAPTER 3

CELL CHARACTERIZATION FOR DELAY

3.1 Introduction

Based on the statistical models and analysis techniques introduced in Chapter 2, a cell library can be constructed in order to analyze process variation effects on circuit performance at a higher level of design abstraction. In this chapter, a detailed description about how to characterize the basic library cell, such as logic gate, for delay analysis due to process variation effects will be given. Firstly, Section 3.2 will discuss which type of SSTA is suitable for constructing a cell library; followed by, in Section 3.3, a review of the corresponding delay models and a discussion about which model is the better choice to employ. Subsequently, the tightness probability based SSTA approach will be introduced in Section 3.4 since this technique provides a great trade-off between modelling accuracy and computational time. Section 3.5 will describe the specific methodology to characterize delay distributions of a library cell considering the different operating conditions. Summary will be outlined at the end of the chapter.

3.2 Why Using Block-Based SSTA?

As introduced in Chapter 2, the SSTA approaches can be sorted into 2 types, the path-based and block-based techniques. The key difference between the two approaches is where in the algorithm the ‘maximum’ function is invoked. The path-based SSTA mainly focuses on finding the critical path within a circuit. It uses a normal-distributed RV to model the distribution of the operating clock frequency of a chip [1, 2], which corresponds to the distribution of critical path delay. Consider the critical path of a circuit; if all the gates in the path are modelled as Gaussian RVs, then the total delay of the path is the sum of these RVs, which can still be expressed as a normal form. Assuming there are *n* gates in a path *P*, the mean delay value is μ_i. Furthermore, consider the delay of each gate to be subject to inter-die and intra-die variations, with a standard deviation σ_{gi} and σ_{ri}, respectively. Since the nominal value of path delay will not change no matter how large the variation is, the mean delay value of path P, μ_p, can be calculated using Equation 3.1:

$$\mu_P = \mu_1 + \mu_2 + \cdots + \mu_n \tag{3.1}$$

It is important to note that, the intra-die variation of a path grows with the increase in the depth of the path in terms of the number of gates. This results from the fact that intra-die variation is a truly random variable component and is independent across gates. Therefore, the standard deviation of the intra-die variation of the path P, $\sigma_{intra,}$ can be expressed as Equation 3.2:

$$\sigma_{intra} = \sqrt{\sigma_{r1}^2 + \sigma_{r2}^2 + \cdots + \sigma_{rn}^2} \tag{3.2}$$

On the other hand, the inter-die variation is a variable shared by all the gates in the same die. Thus the standard deviation of the inter-die variation of path P, σ_{inter}, can be expressed as Equation 3.3:

$$\sigma_{inter} = \sigma_{g1} + \sigma_{g2} \dots + \sigma_{gn} \tag{3.3}$$

Based on Equations 3.2 and 3.3, it can be concluded that the contribution of intra-die variation to the total path variation will decrease with the increase of the path depth. Assuming all the gates along path P have the same standard deviations for inter-die and intra-die, σ_g and σ_r, then:

$$\frac{\sigma_{inra}}{\sigma_{inter}} = \frac{\sigma_r}{\sqrt{n}\sigma_g} \tag{3.4}$$

In a large circuit, normally there will be multiple paths which are expected to have a significant probability of becoming critical and strongly influence the overall delay performance. The goal of path-based SSTA is to estimate the ‘maximum’ of a selected set of critical paths in order to compute circuit delay PDF, which is a crucial step in SSTA. The ‘minimum’ operation is also needed for the computation of the shortest path delay distribution. However, it can be derived from the ‘maximum’ operation.

Figure 3-1 shows a general view of performing a path-based SSTA for a given circuit. It is a depth-first traversal of the timing graph. The basic advantage of this approach is its low complexity, since the analysis is clearly split into two parts: the computation of path delays and one statistical maximum operation. Hence, much of the initial research into SSTA was focussed on path-based approaches [3-9].

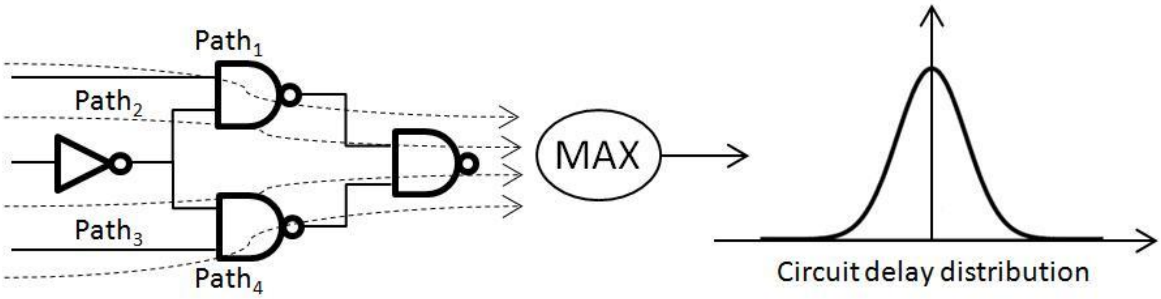


Figure 3-1 General view of path-based SSTA.

The major difficulty of the path-based approaches is that there is no efficient algorithm available to establish critical paths in a circuit. It is unclear how to select the initial set of paths before performing SSTA. Even though path-based approaches provide a simplified statistical computation since only the ‘add’ operation is executed during analysis, the complexity of the pre-analysis work is actually huge. Additionally, for a large circuit, the number of paths that must be considered can be very high. This makes the path selection problem even more complicated. On the other hand, the lower computational complexity of the path-based SSTA is made under the assumption that all the critical path delay distributions are independent of each other. If taking correlations into account the analysis tends to lose its computational efficiency. Therefore, most of the later research has focused on the block-based approaches. Most importantly, if considering using a cell library to analyse circuit delay performance, library cells must be characterized before being used in the circuit. When constructing the cells, there is no information available about the structure of the circuit being designed and whether or not it involves a critical path. Thus, it is quite difficult to characterise circuit cells which will be used in path-based SSTA. Therefore, the block based SSTA becomes a better delay PDF propagating algorithm for building a statistical cell library.

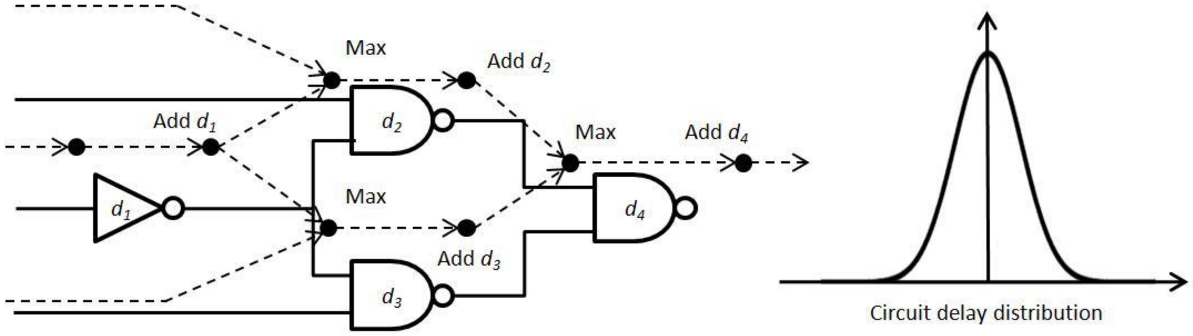


Figure 3-2 General view of block-based SSTA.

The block-based SSTA is closer to the traditional STA algorithm which propagates the delay PDF through a circuit in a topological manner. It is a breadth first traversal of the timing graph. The general view of the block-based SSTA approach is shown in Figure 3-2. Two types of signal arrival times (rise and fall) will be propagated at each node in a circuit, resulting in a runtime that is linear with circuit size. As described in Chapter 2, the timing distributions at each node are computed using two basic operations: addition and maximum. There is no difficulty in summing two variables; however, as mentioned in previous section, computing the statistical maximum of two correlated arrival times is complicated. Obviously, performing a block-based SSTA in a circuit requires the 'maximum' operation more frequently compared to a path-based approach. This will lead to an increase in the computational complexity. However, block-based approaches do not require establishing critical paths before performing the analysis, therefore, its computational time is more predictable than the path-based approaches. Due to its runtime advantage, many current research and commercial efforts have adopted the block-based approach. Furthermore, block-based SSTA lends itself to an incremental analysis, which is a huge advantage not only for the characterization of library cells in a hierarchical manner, but also diagnostic and optimization applications. Under these conditions, the block-based SSTA is employed for constructing the statistical cell library.

3.3 Delay Models for Block-Based SSTA

Having established the type of delay analysis algorithm to be used, the corresponding models need to be discussed. There are a number of models available to capture the timing characteristics of circuits. As discussed in Chapter 2, the signal arrival time and gate delay time are modeled as worst case and corners in traditional static timing analysis (STA). When the timing analysis merges into the statistical domain, the research efforts are focused on directly representing gate delays with RVs characterized by their distributions or statistical characteristics [10]. This section will introduce the different gate delay models available for SSTA, followed by a discussion about the proper model to construct the library cells for the analysis of process variation effects at a higher level of abstraction.

3.3.1 Discrete delay models

In order to handle probability distributions in SSTA, the first effort was made by L. Jing-Jia, et al [11] who proposed a model using discrete PDFs to represent the delay variation. Similar approaches are also proposed in [12, 13]. This technique performs SSTA in a computationally deterministic fashion rather than random sampling based approaches, such as Monte Carlo simulation. The gate delay PDF is generated by sampling a continuous distribution with a user-defined sampling step as shown in Figure 3-3.

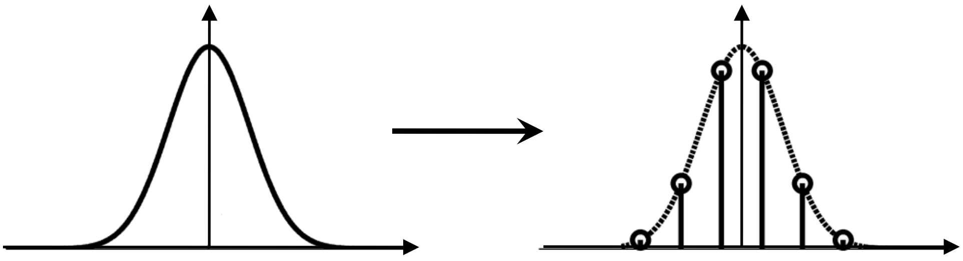


Figure 3-3 Sampling a continuous PDF of delay to generate a discrete PDF.

The continuous delay PDF is assumed to be given and could be pre-generated by the Monte Carlo technique before the sampling. The discrete PDF needs to be renormalized after sampling to ensure that the sum of the probability pulses is equal to one. The sampling step provides a trade-off in terms of computational time and modeling accuracy. If the sampling step is small, the shape of the discrete PDF will be very close to the original distribution. However, the large number of samples in the model will increase the computational complexity of SSTA. A larger sampling step will speed up the analysis but lose accuracy. If the sampling window is larger than the width of the delay PDF, it becomes the worst case model. Thus, choosing the sampling step to generate discrete PDF is always a difficult and tricky procedure.

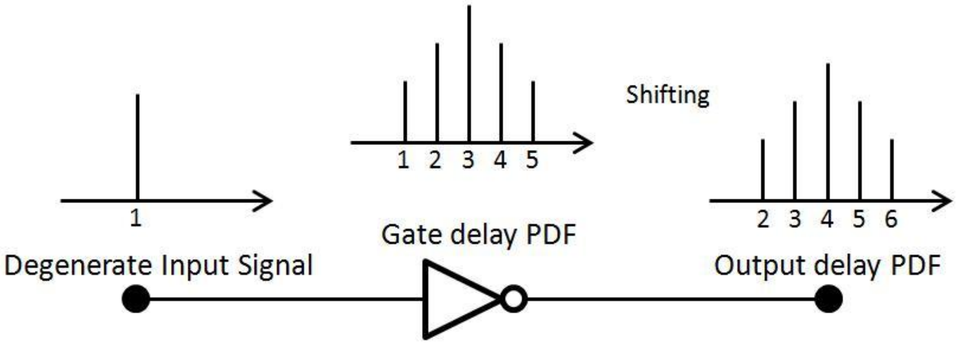


Figure 3-4 Shifting gate delay PDF by degenerate input signal delay.

When performing an addition operation in a timing analysis using the discrete delay model, the output signal distribution is obtained by simply shifting the gate delay distribution by the input delay. Figure 3-4 shows how to propagate the delay distribution through an inverter circuit when the input signal is degenerate, that is when the signal is primary input and without variation. The numbers on the x-axis represent the delay value associated with the particular discrete probability distribution sample. However, in the case where the input signal delay is non-degenerate, a set of shifted output PDFs will be generated as shown in Figure 3-5.

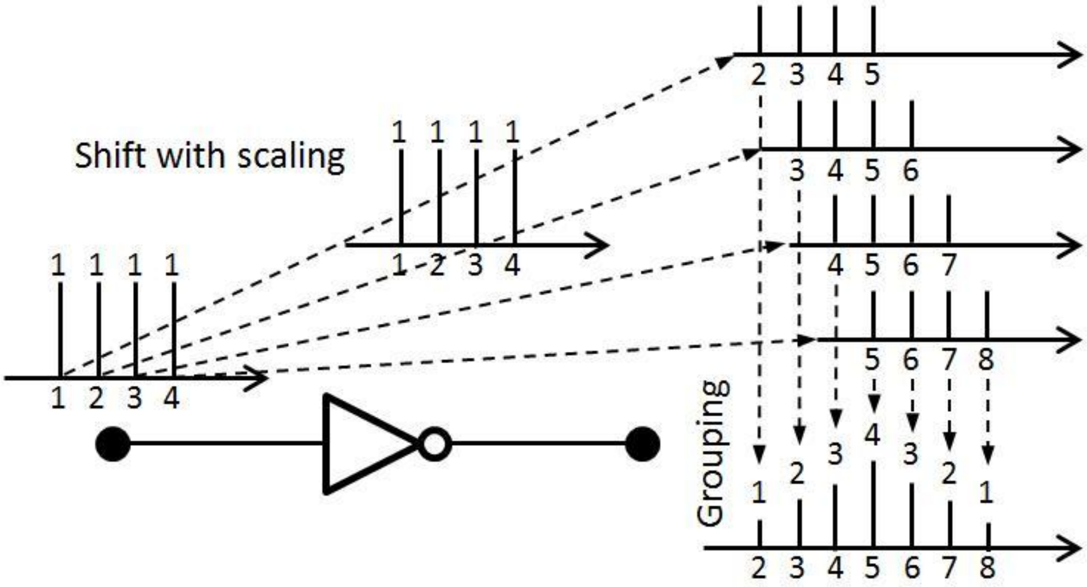


Figure 3-5 Shifting gate delay PDF by non-degenerate input signal delay.

Each of the shifted delay PDFs corresponds to a discrete event in the input signal PDF. The final output delay distribution is obtained by combining these shifted PDFs using Bayes' theorem [14]. The gate delay PDFs need to be scaled by a factor which is the probability of the input signal event occurring before the shifting. Subsequently, all the shifted discrete distributions will be grouped by summing their probabilities at each time point. It needs to be noted that, the sum of all the probability events in a PDF should be equal to 1. Therefore the actual probability of an event in a PDF will be computed by dividing its total value by the sum of the numbers corresponding to all the events in the same PDF. The overall computation can be expressed as Equation 3.5:

$$f_s(t) \ = \sum_{i=-\infty}^{\infty} f_x(i) f_y(i-t) = \ f_x(t) * f_y(t) \tag{3.5}$$

The operator "*" represents convolution and "*f*" represents the PDF of the corresponding RV. When performing an addition of 2 discrete RVs *x* and *y*, the sum, *s* = *x* + *y*, can be expressed as a convolution of their PDFs.

The statistical maximum of two RVs modeled in discrete form can be computed using Equation 3.6 [15, 16]:

$$f_z(t) = F_x(t)f_x(t) + F_y(t)f_y(t) \tag{3.6}$$

Where $z = max(x, y)$, and F represents the cumulative distribution function (CDF) of the corresponding RV. The two RVs x and y are assumed to be independent of each other.

Based on the Equations 3.5 and 3.6, each multiplication in the convolution and *max* computation results in a quadratic function, generating a total computational complexity of $O(n^2)$ [1], where n is the number of events in the discrete delay model. This makes this modeling technique less feasible to be applied in a large circuit. On the other hand, Equations 3.5 and 3.6 are only valid under the condition that the processing RVs are independent of each other. The modeling of inter-die and intra-die variations is totally ignored. Furthermore, the discrete modeling technique assumes the gate delay distribution is already known before sampling, which requires extra simulation runs to compute the actual PDFs of the gate. Therefore, the model characterization work becomes too cumbersome. Most importantly, the gate delay distributions could be significantly different when considering different variational sources and the amount of deviation of each RV. Consequently, it may be needed to model each delay PDF under different variation conditions using discrete models, which makes the analysis process even more complicated. Consequently, a more efficient and feasible model is needed for constructing the cell library.

3.3.2 Canonical delay model

In recent research work into SSTA, the canonical gate delay form becomes more and more popular and has been used in many SSTA approaches [17-20]. This modeling technique use RVs to represents device parameter variations, such as gate length and oxide thickness, rather than the total gate delay distribution in discrete models. For most of the cases, the RVs are assumed to be Gaussian and each RV of the corresponding parameter can be divided into 2 components: inter-die and intra-die variations. Therefore it only needs 3 parameters to represent the normal-distributed RV: the expected or nominal parameter value (mean value) and the 2 user-defined standard deviations for global and random variation components respectively as shown in Equation 3.7.

$$\Delta P = P_{nom} + \Delta P_{inter} + \Delta P_{intra} \tag{3.7}$$

The Gaussian approximation for delay is based on the assumption that variations in the process parameters are typically small and their impact on gate/circuit delay is linear. The gate delay distribution can be obtained by the weighted addition of these device parameter RVs. Each RV will be multiplied by a sensitivity factor to move the variation effects from the device level to gate level. Additionally, the environmental sources of variation can be modeled in the same fashion. Assuming the variation in parameter *x* will cause the gate delay d to deviated by its mean value, and *x* is a Gaussian RV with a normal value μ_x and standard deviation σ_x. Therefore, d = *f(x)*. The sensitivity factor β of *x* with respect to *d* can be computed using Equation 3.8:

$$\beta = \frac{f(\mu_x + \sigma_x) - f(\mu_x - \sigma_x)}{2\sigma_x} \tag{3.8}$$

Now the only unknown factor in the expression above is the function *f* whose complexity will directly affect the further timing analysis efficiency. The common solution to derive *f* is using SPICE simulation runs to find the response delay performance with different values of variational source. Figure 3-6 shows an example for computing the sensitivity factor for the transistor gate length L_{eff} with respect to the inverter circuit fall time delay using SPICE simulations.

In Figure 3-6, *V(in)* is the input signal to the inverter circuit; *V(out)1* is the output signal when L_{eff} deviates to its one sigma value and *V(out)2* is the output signal when L_{eff} deviates to its negative one sigma value; *a* is the timing point when *V(in)* drops down to 50% of V_{dd} value; b_1 and b_2 are the timing points when *V(out)1* and *V(out)2* also drop down to 50% of V_{dd} value respectively. The two delay responses required in Equation 3.8 can be measured as (b_1-a) and (b_2-a). Therefore, the sensitivity factor for L_{eff} can be simply obtained using two SPICE runs.

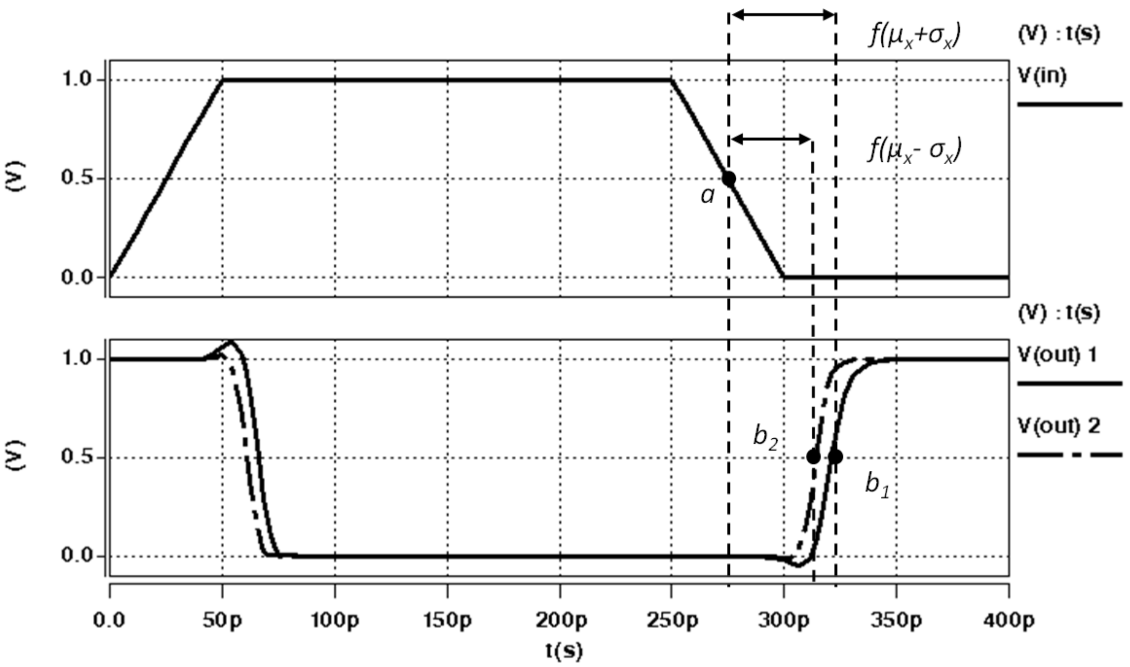


Figure 3-6 Sensitivity analysis for gate channel length.

For a given variation source, the sensitivity factor is the same for both inter-die (global) and intra-die (random) components. Thus the gate delay distribution can be expressed as Equation 3.9:

$$D = \mu_D + \sum_{i=1}^{n} \beta_i \Delta P_{inter,i} + \sum_{i=1}^{n} \beta_i \Delta P_{intra,i} \tag{3.9}$$

where *n* is the total number of the RVs in the model. $\Delta P_{intra,i}$ can be combined using Equation 3.2 which is introduced in Section 3.2. Let G_i be inter-die component i^{th} variation source and *R* be combined intra-die variable with a new sensitivity factor β_{n+1}, then the final expression for the gate delay polynomial is shown in Equation 3.10, which is called the canonical model [20-22].

$$D(Delay) = \mu_D + \sum_{i=1}^{n} \beta_{Di} G_i + \beta_{D(n+1)} R \tag{3.10}$$

The canonical gate delay model is in the form of a 1st order variable polynomial. It is simple to characterize by SPICE based simulation and easy to apply to the block-based timing analysis. That is why it has been widely used in most of the SSTA approaches.

However, the canonical model has been criticized for its accuracy especially when the parameter variation is huge and the delay distribution response tends to be non-linear. As a potential solution, high-order model has been proposed as described in the following section.

3.3.3 Quadratic delay model

If the gate delay's dependency on the global variation sources is nonlinear, Taylor series could be a potential solution to analyze such a nonlinear function systematically [23]. In mathematics, a Taylor series is a representation of a function as an infinite sum of terms that are calculated from the values of the function's derivatives at a single point [24]. Let G_1, G_2, … ,G_n be the n standard Gaussian RVs with zero mean and unity variance, the Taylor expansion of the delay distribution can be expressed as equation 3.11:

$$D(G_1 \ldots G_n) = m + \alpha R + \sum_{j=1}^{\infty} \left\{ \frac{1}{j!} \left(\sum_{i=1}^{n} G_i \frac{\partial}{\partial G_i'} \right)^j \times D(G_1' \ldots G_n') \right\}_{G_1' = \cdots = G_n' = 0} \tag{3.11}$$

where R is the local variation and m is the delay value if no variation has occurred, m=D(0, 0, … , 0). If the Taylor expansion is truncated at the first order, Equation 3.10 becomes the canonical form, and the value of m is the mean value of delay distribution (μ_D). However, for the higher order model, m may not be equal to μ_D. Since the local variable R represents the overall effect of all the localized variations, it is normally assumed to be Gaussian according to the "law of large numbers [25]."

Obviously the accuracy of the model can be improved by increasing the order of the Taylor expansion but at a penalty of computational cost. A reasonable trade-off has to be made. In [23] the author states that, based on their experiments, for parameter variation up to 30% of the nominal value ($3\sigma < 30\%$) the 1st order canonical expression can maintain the accuracy for modelling delay variation. If the parameter variation is larger than 30%, the modelling error of canonical form becomes unreasonable. Therefore a quadratic delay model has been proposed in order to analyse large process variation effects efficiently [23].

In the quadratic model, the gate delay D is a nonlinear function of the global variations. The Taylor expansion in Equation 3.1 will be truncated up to the second order as Equation 3.12:

$$D \approx m + \alpha R + \frac{\partial D}{\partial L}L + \frac{\partial D}{\partial V}V + \cdots + \frac{1}{2}\frac{\partial^2 D}{\partial L^2}L^2 + \frac{\partial^2 D}{\partial L \partial V}LV + \frac{1}{2}\frac{\partial^2 D}{\partial V^2}V^2 \ldots \tag{3.12}$$

where m is a constant and L, V … are the global variations. The variational parameters are pre-defined before analysis. The coefficients in this Taylor expansion can be analytically extracted from the designed SPICE simulations using the finite difference method, just like the sensitivity analysis used in characterizing the canonical model. After fitting all the coefficients for the corresponding variables, the Taylor expansion can be re-expressed as Equation 3.13, which is called quadratic gate delay model [23, 26]. The full expression of quadratic model has been described in Chapter2, Section 2.8.1.

$$D(Delay) = \mu_D + {\beta_g}^{*}\delta_g + {\delta_g}^{*}\Gamma_g\delta_g + \beta_{D(n+1)}R \tag{3.13}$$

Figure 3-7 shows the CDF and PDF plots of an example inverter circuit with three different modeling techniques, Monte Carlo, canonical and quadratic. The parameter variations are set to 30% of their nominal value.

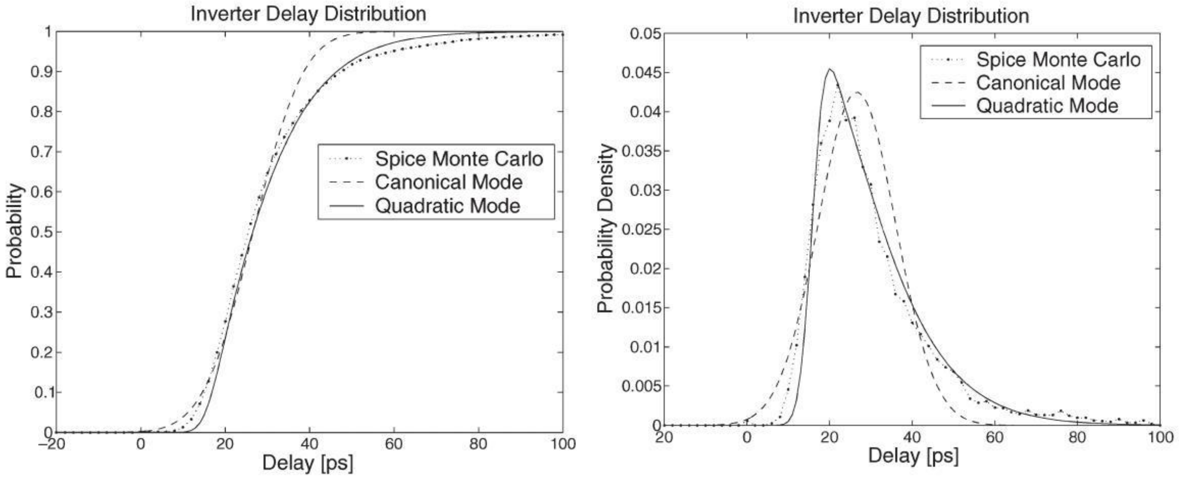


Figure 3-7 Inverter delay CDFs and PDFs with parameter variation σ/μ=30% [23].

The Monte Carlo analysis result is the closest prediction to the real circuit delay distribution. Therefore it is normally used as a reference to compare the modeling accuracy of the other techniques. From the graphs above, it can be observed that the quadratic model captures more delay characteristics of the inverter when the parameter variations are large.

3.3.4 Why use the canonical model?

In order to perform a timing analysis using the quadratic delay model, it requires 'addition' and 'maximum' operations using second-order polynomial expressions. However, the complexity of both the polynomial coefficient fitting and delay distribution calculation using the quadratic model grows exponentially with the increase in the number of variation sources. According to[27], there are approximately 5 to 10 sensitive process parameters under variation, which have significant effects on circuit performance, for each type of transistor as present. Thus it is too time-consuming to implement the quadratic delay model in a very large circuit, and less feasible to construct a cell library using the higher order delay models which take a large number of simulations runs to characterize a cell.

On the other hand, according to the 90nm technology parameter variations outlined in the ITRS roadmap [28], only a few parameters such as the effective channel length has 3 sigma value much greater than 30% of their mean values, some other delay sensitive parameters such as threshold voltage V_{th}, supply voltage V_{dd} and gate oxide thickness T_{ox} are only 15% of average variation. Consequently the actual delay distribution of a circuit should show much more linearity. Table 5 lists the roadmap variations for the device parameters which have most impact on the circuit performance.

Table 5 Technology road map.

Parameter	**Variations (3σ/μ)**				
Year	**1997**	**1999**	**2002**	**2005**	**2006**
L_{eff}	32%	33%	38%	40%	47%
W_{eff}	25%	30%	28%	30%	26%
V_{dd}	10%	10%	10%	10%	10%
V_{th}	10%	10%	10%	11%	13%
T_{ox}	8%	8%	9%	12%	16%

In order to illustrate the characteristics of the circuit delay distribution under reasonable variations, Figure 3-8, as an example, shows the delay probability density function (PDF) of an inverter circuit with the ITRS 90nm technology variation specification. The histogram is generated by Monte Carlo simulation and the solid line represents the first order polynomial fitting graph.

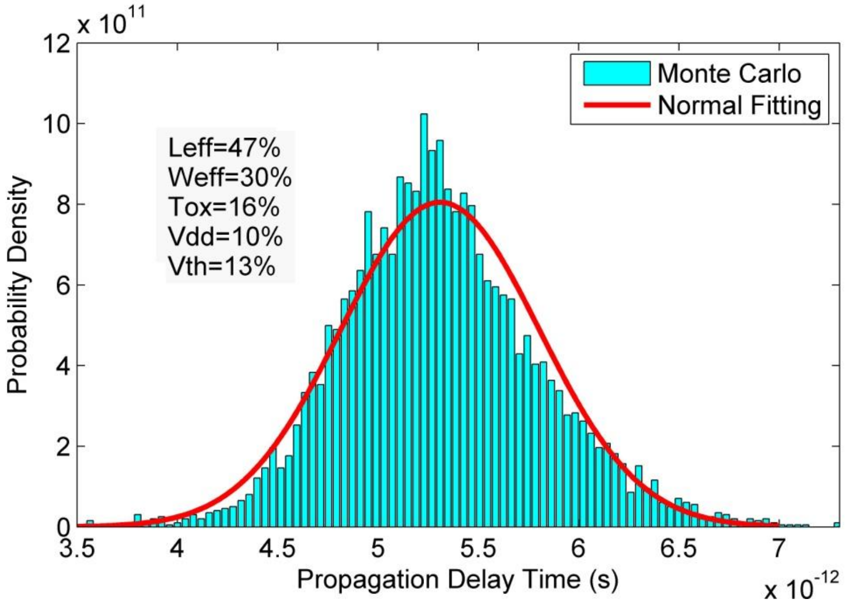


Figure 3-8 Inverter delay PDFs of 90nm technology.

It can be observed from the graph above, the 2 PDFs are well matched. It indicates that the delay distribution shows a lot of normality and is quite close to the Gaussian distribution. Thus the first order canonical delay model is sufficiently accurate to capture the effects of process variations on delay even with a small number of highly variational parameters. On the other hand, the first order polynomial form of delay distribution representation can save a significant amount of modelling and computational time over the quadratic expressions, making it more feasible for use in implementing a cell library. Under these conditions, first order canonical gate delay model becomes the better choice to use.

3.4 Tightness Probability Based SSTA

As discussed in the previous section, the first-order canonical model will be employed for each cell in a circuit which considers both global and local components of variation. In performing a timing analysis for a given circuit, the delay distributions are calculated for each active signal path from the primary inputs to the primary outputs in a circuit using block-based SSTA. The main difficulty with the canonical model based SSTA is focussed on how to re-express the non-normal statistical maximum result into a canonical form again so that the delay distribution can be propagated through the circuit. C. Visweswariah et al [20] have proposed a tightness probability based approach which becomes one of the most popular solutions for SSTA using 1st order canonical model, because of its computational efficiency. In this section, the concept of tightness probability will be introduced first, followed by a description of the key idea of the timing analysis approach.

3.4.1 Concept of tightness probability

Tightness is also called "binding probability" [29]. For two given variables *A* and *B*, the tightness probability T_A of *A* is the probability that it is larger than or dominates *B*, and T_B = (1- T_A). If given *n* variables, then the tightness probability of each variable is the probability that it is larger than or dominates all the others [20]. Assuming *A* and *B* are in the 1st order canonical form as shown in Equations 3.14 and 3.15:

$$A = \mu_A + \sum_{i=1}^{n} \beta_{Ai} G_i + \beta_{A(n+1)} R \tag{3.14}$$

$$B = \mu_B + \sum_{i=1}^{n} \beta_{Bi} G_i + \beta_{B(n+1)} R \tag{3.15}$$

Then the covariance matrix of *A* and *B* can be expressed as Equation 3.16. The variance of a variable is a measure of the dispersion of the values taken by the variable around its mean value, and the covariance matrix generalizes the concept of variance to multiple dimensions.

$$Cov(A,B) = \begin{bmatrix} \beta_{A1} & \beta_{A2} & \cdots & \beta_{An} & \beta_{A(n+1)} & 0 \\ \beta_{B1} & \beta_{B2} & \cdots & \beta_{Bn} & 0 & \beta_{B(n+1)} \end{bmatrix} [V] \begin{bmatrix} \beta_{A1} & \beta_{B1} \\ \beta_{A2} & \beta_{B2} \\ \vdots & \vdots \\ \beta_{An} & \beta_{Bn} \\ \beta_{A(n+1)} & 0 \\ 0 & \beta_{B(n+1)} \end{bmatrix} \tag{3.16}$$

where *V* is the *(n+1)*×*(n+1)* covariance matrix of all the variables of the selected parameters in canonical forms. Assuming all the *(n+1)* variables in each canonical model are independent, then *V* becomes a unity matrix. Thus Equation 3.16 can be simplified into Equation 3.17:

$$Cov(A,B) = \begin{bmatrix} \sum_{i=1}^{n+1} \beta_{Ai}^2 & \sum_{i=1}^{n} \beta_{Ai}\beta_{Bi} \\ \sum_{i=1}^{n} \beta_{Ai}\beta_{Bi} & \sum_{i=1}^{n+1} \beta_{Bi}^2 \end{bmatrix} = \begin{bmatrix} \sigma_A^2 & \rho\sigma_A\sigma_B \\ \rho\sigma_A\sigma_B & \sigma_B^2 \end{bmatrix} \tag{3.17}$$

where σ_A and σ_B are the standard deviations of *A* and *B* which can be computed by matching the two 2×2 matrices in Equation 3.17. The value of the correlation coefficient ρ can be derived in the same fashion.

Let:

$$\Phi(y) \equiv \int_{-\infty}^{y} \varphi(x)dx \tag{3.18}$$

$$\varphi(x) \equiv \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \tag{3.19}$$

$$\theta \equiv (\sigma_b^2 + \sigma_c^2 - 2\rho\sigma_b\sigma_c)^{\frac{1}{2}} \tag{3.20}$$

Then the tightness probability T_A can be expressed as Equation 3.21:

$$T_A = \int_{-\infty}^{\infty} \frac{1}{\sigma_A} \varphi\left(\frac{x-\mu_A}{\sigma_A}\right) \Phi\left[\frac{\left(\frac{x-\mu_B}{\sigma_B}\right) - \rho\left(\frac{x-\mu_A}{\sigma_A}\right)}{\sqrt{1-\rho^2}}\right] dx = \Phi\left(\frac{\mu_A - \mu_B}{\theta}\right) \tag{3.21}$$

Based on the expression of tightness probability, the analytical solutions for the first two moments (mean and variance) of the statistical maximum operation, which has been introduced in Chapter 2 Section 2.8.2, can be re-expressed as Equations 3.22 and 3.23 [30, 31]:

$$E[Max(A,B)] = \mu_A T_A + \mu_B(1 - T_A) + \theta\varphi\left[\frac{\mu_A - \mu_B}{\theta}\right] \tag{3.22}$$

$$Var[Ma\,x(A,B)] = \\ (\sigma_A^2 + \mu_A^2)T_A + (\sigma_B^2 + \mu_B^2)(1 - T_A) + (\mu_A + \mu_B)\theta\varphi\left[\frac{\mu_A - \mu_B}{\theta}\right] - \{E[Max\,(A,B)]\}^2 \tag{3.23}$$

Therefore, with the concept of tightness probability the expected value and variance of the statistical maximum operation *Max(A, B)* can be computed analytically and efficiently. The statistical minimum result can be derived by *Min(A, B)= −Max(−A, −B)*. Therefore, in this thesis and most of the proposed papers about SSTA, only the details of *Max* operation are stated. The time for computing *E[Max(A, B)]* and *Var[Max(A, B)]* is linear with the number of variation sources.

3.4.2 Application of tightness probability in SSTA

In block-based SSTA, the Gaussian delay PDFs in the 1st order canonical form are propagated through the circuit by estimating the distributions at each node. The crucial step is to maintain the node delay PDFs, which are calculated by the statistical “Add” or “Max” operation, in the same canonical form. Thus the SSTA can be kept alive traversing the timing graph. As discussed in the previous section, the sum of multiple normal variables is still a normal variable. Therefore, performing a statistical “Add” operation for canonical models is straightforward. However, the “Max” result of Gaussian variables is no longer Gaussian, whose shape is more like a normal distribution but with a skewness. Consequently, the “Max” result can only be approximated in order to keep it in canonical form, where the tightness probability concept can be applied.

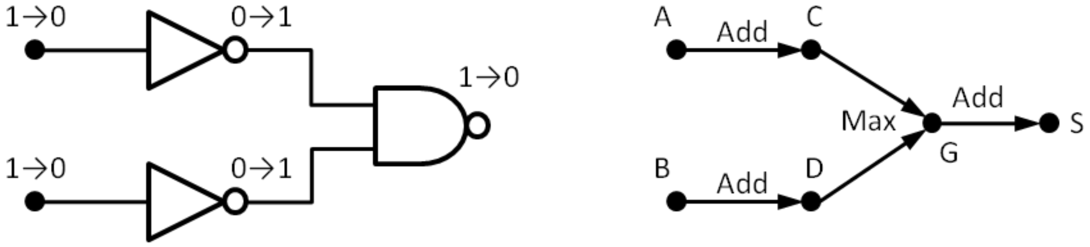


Figure 3-9 An example circuit and the corresponding timing graph to illustrate the "Add" and "Max" operations in SSTA.

Figure 3-1 shows an example circuit and its corresponding timing graph. The two falling input signal of the circuit will cause a rising signal transition at the output. In order to perform a block-based SSTA, it requires both the statistical "Add" and "Max" operations performed on the signal transition case shown in the above figure. The distributions of all the input signal arrival times and gate delay times are modeled as 1st order canonical forms. The polynomial at node *G* can be derived using Equation 3.24:

$$\begin{aligned} G = \;& Max[\{A + C\}, \{B + D\}] \\ = \;& Max\left[\left\{(\mu_A + \mu_C) + \sum_{i=1}^{n} (\beta_{Ai} + \beta_{Ci})\, G_i + \left(\sqrt{\beta_{A(n+1)}^2 + \beta_{C(n+1)}^2}\right) R\right\}\right. \\ & \left. , \left\{(\mu_B + \mu_D) + \sum_{i=1}^{n} (\beta_{Bi} + \beta_{Di})\, G_i + \left(\sqrt{\beta_{B(n+1)}^2 + \beta_{D(n+1)}^2}\right) R\right\}\right] \\ = \;& Max\left[\left\{\mu_X + \sum_{i=1}^{n} \beta_{Xi}\, G_i + \beta_{X(n+1)} R\right\}, \left\{\mu_Y + \sum_{i=1}^{n} \beta_{Yi}\, G_i + \beta_{Y(n+1)} R\right\}\right] \end{aligned} \tag{3.24}$$

where *A* and *B* are the variables for the input signals of the circuit shown in Figure 3-9, *B* and *D* represent the delay distributions of the two inverters, *X* and *Y* indicate the summed polynomials of *(A+C)* and *(B+D)*. The local variables are treated in a root of the sum of the squares (RSS) fashion in the "Add" operation, because of their independent randomness. As discussed in the beginning of this chapter, the local variation reduces the spread of delay of a long path consisting of many stages.

The variable summing process has been demonstrated in Equation 3.24. The next step is to find the "Maximum" of the *X* and *Y*. In traditional static timing analysis, when multiple signals converge to the node *G*, the one with larger delay value will pass through for all downstream purposes, the characteristics of the dominant potential arrival time determines the arrival time at G, and all the other potential arrival times are ignored. It is as if the slowest arrival signal has a tightness probability of 100%, the tightness probability for others is 0%. When the delay model moves into the probabilistic domain, the characteristics of the arrival time at *G* can be expressed from *X* and *Y* in the proportion of their tightness probabilities. For example, if T_X*=0.6* and T_Y*=0.4,* then the delay distribution at G can be computed by a weighted-sum of *X* and *Y* with a 3:2 sensitivity ratio. Therefore, the sensitivities of the global variations of *Max(X, Y)* can be computed from Equation 3.25:

$$\beta_{Gi} = T_X\beta_{Xi} + (1 - T_X)\beta_{Yi} \tag{3.25}$$

The mean value of *Max(X, Y)* can be derived by Clark's analytical solution in Equation 3.22, the only remaining part of *Max(X, Y)* is the sensitivity factor for the local RV, $\beta_{G(n+1)}$. Since local RVs are combined in an RSS fashion which is not a linear function, it cannot be computed using Equation 3.25. The way to calculate $\beta_{G(n+1)}$ is to find a value which makes the variance of the fitted "Max" result equal to the variance of the analytical "Max" result which can be obtained using Equation 3.23. It was shown that a valid value of $\beta_{G(n+1)}$ always exists as the residue $\left(\sigma_G^2 - \sum_{i=1}^{n}\beta_{Gi}^2\right)$ is always greater than or equal to 0. Now the result of *Max(X, Y)* has been re-expressed in its canonical form again. When there are more than 2 timing edges converging at a node, only two of them are "*Maxed*" at a time, then the result will be used to perform the next "*max*" operation with other timing variables, and so on.

The key idea of the tightness probability based SSTA has been described in this section. According to the literature [10, 20], this approach can effectively compute the first two moments of a non-linear "Max" operation result in SSTA, and maintain an acceptable error rate within 5%. The major computational complexity trade off of tightness probability based SSTA makes it one of the most popular timing analysis techniques using 1st order canonical delay model. That is also the main reason why this approach is employed in the proposed cell library. Having established the delay modeling and analysis techniques, the statistical cell library can be constructed on this basis.

3.5 Cell Characterization in Different Operating Conditions

In this section a description of the characterization of the process variation effects on the delay in a library cell will be outlined. The 1st order canonical delay model can capture the gate delay uncertainty caused by process variations. However, the delay distributions of a gate under the same variation specifications, will behave differently under different circuit operating conditions.

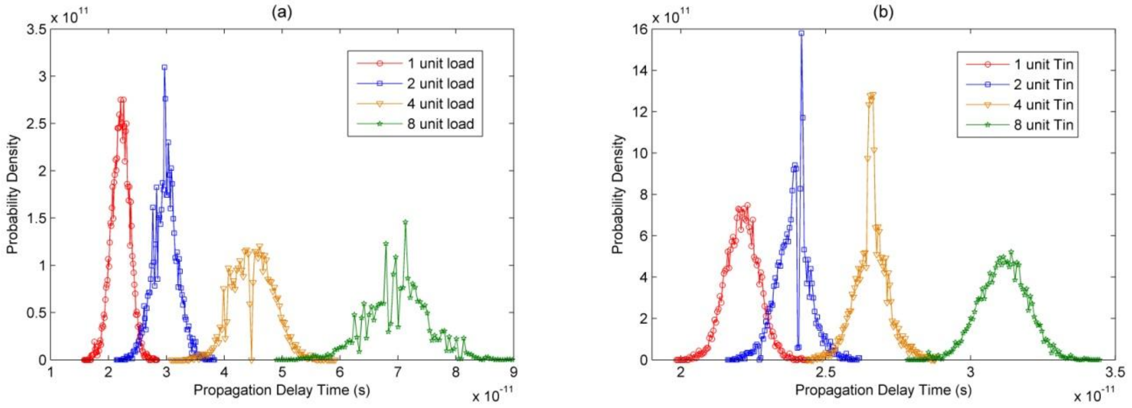


Figure 3-10 Inverter delay PDFs with different T_{in} and C_L.

The major circuit factors which cause this difference in the delay PDFs are the gate input signal transition time T_{in} and the output load capacitance C_L. In order to illustrate the effects of T_{in} and C_L, a number of delay PDFs for an example inverter circuit are shown in Figure 3-10 for comparison purposes. The inverter gate is under the variation effect from L_{eff} with a sigma value equal to 10% of the mean. The distributions in Figure 3-10 (a) are measured with the same T_{in} but different C_L conditions, the distributions in Figure 3-10 (b) are measured with a same C_L but different T_{in} values. It is observed that the delay PDFs show significant difference in each case.

It is very difficult to model the operating condition effects (T_{in} and C_L) on propagation delays expressed in canonical form, typically the table look-up approach will solve this problem [17], where the delay time is sampled with respect to a wide range of C_L and T_{in} values, then saved in memory. A huge number of delay samples are required to model one gate delay in order to cope with every value of C_L and T_{in}, which makes the library cell characterization onerous and inefficient. Additionally, the huge amount of data associated with a single cell model also makes the whole cell library very costly in terms of memory space. Simplified tables are desperately needed in order to increase the practical applicability of the statistical cell library.

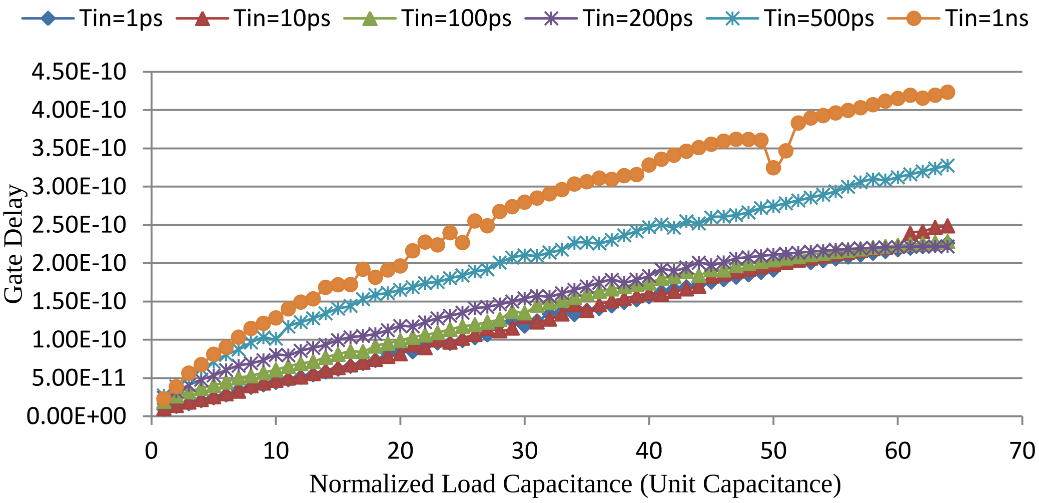


Figure 3-11 Inverter delay versus C_L responses.

In order to investigate the relationship between the gate delay characteristic and the circuit operating conditions, a number of inverter delay versus C_L responses were plotted as shown in Figure 3-11. Similarly, the inverter delay versus T_{in} responses graphs are shown in Figure 3-12.

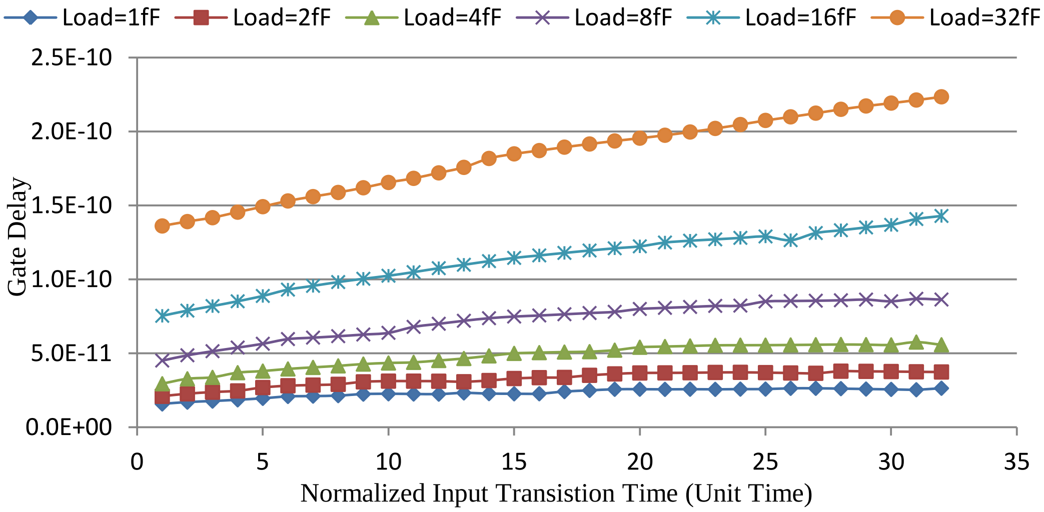


Figure 3-12 Inverter delay versus T_{in} responses.

Based on the observations from Figure 3-11 and Figure 3-12, the overall graphs are smooth, which means gate delay has a close-to-linear relationship with C_L and T_{in}. For this reason, a piecewise linear function can be used to fit the delay samples for different drive and load conditions in order to simplify these tables. Only a few delay values are sampled as break points at some typical values of C_L and T_{in}, any delay values in close proximity to these will be estimated from the linear function of its adjacent break points.

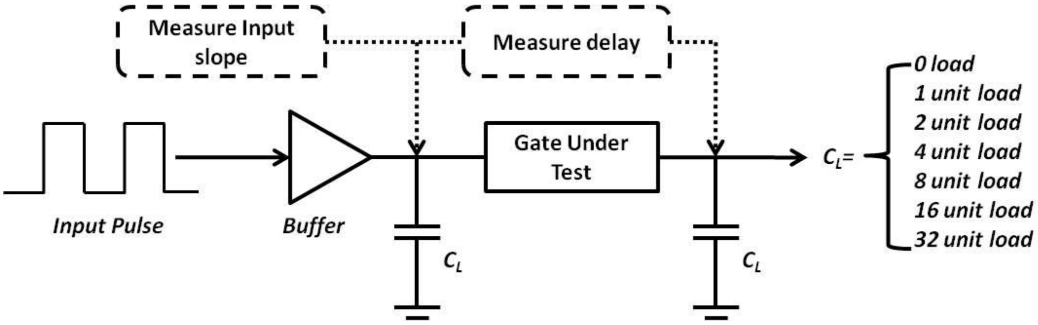


Figure 3-13 Sample delay breaking point.

Figure 3-13 shows how to characterize the simplified look-up table for basic cells. The buffer at the input node of gate under test will provide a realistic input signal slope. The gate delay time will be sampled at 7 output load values as shown in Figure 3-13, where the unit load means the static input capacitance of an inverter circuit. The load capacitance of a gate can be approximated as the input capacitance of the successive gates, as shown in Figure 3-16. In CMOS circuits, the value of C_L at any node in a circuit can be a multiple of the unit load [32, 33].

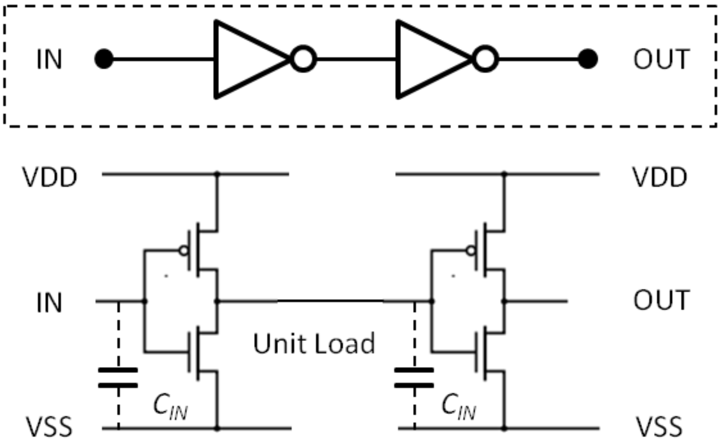


Figure 3-14 Schematization of two cascade inverters.

The existence of the device parameter variations will also affect the input capacitance value of gate C_{in}, which means the potential gate load capacitance value could be a variable. Fortunately, it was found that the C_{in} value variation of static logic gates is typically very small. Based on the experimental results of the extensive Monte Carlo simulations on several gates with different sizes in a 90nm CMOS technology [32, 33], the relative variation σ/μ of C_{in} is within 0.5% under both the inter-die and intra-die variations. Figure 3-15 shows the statistical distribution of the static input capacitance values of an inverter circuit, the sigma-to-mean ratio is 0.5%. Interestingly, the relative variation for larger-sized gates than inverter was found to be even lower. Therefore, the C_{in} values of logic gates can be treated as constants when modeling the circuit operating condition effects on cell delay.

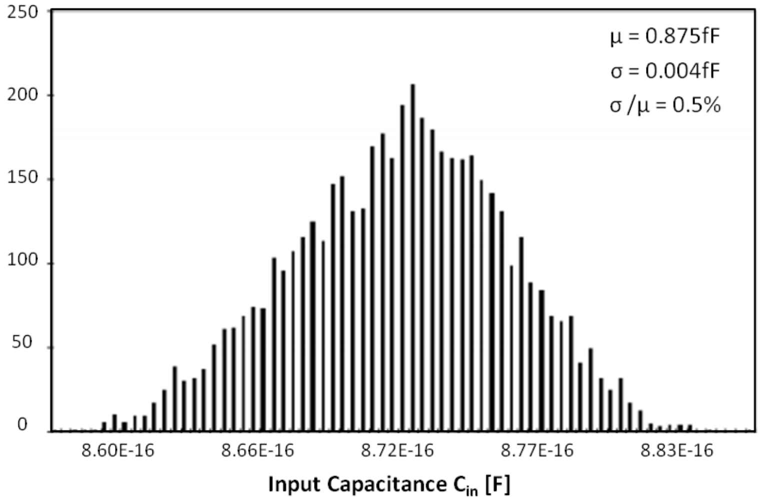


Figure 3-15 Statistical distribution of the input capacitance values of an inverter.

On the other hand, the input signal slope of a gate cannot be easily controlled in a real circuit, so that the input signal slope of a test vector in Figure 3-13 is obtained by adjusting the capacitance values at the input node of the gate (using the same 7 capacitance values). It, therefore, needs 49 simulation runs to build the look-up table for gate delay, which are the mean values in canonical delay form. Figure 3-16 (a) shows the relationship between propagation delay and operating conditions for the inverter circuit, and Figure 3-16 (b) shows the result of inverter delay fitting using the simplified tables.

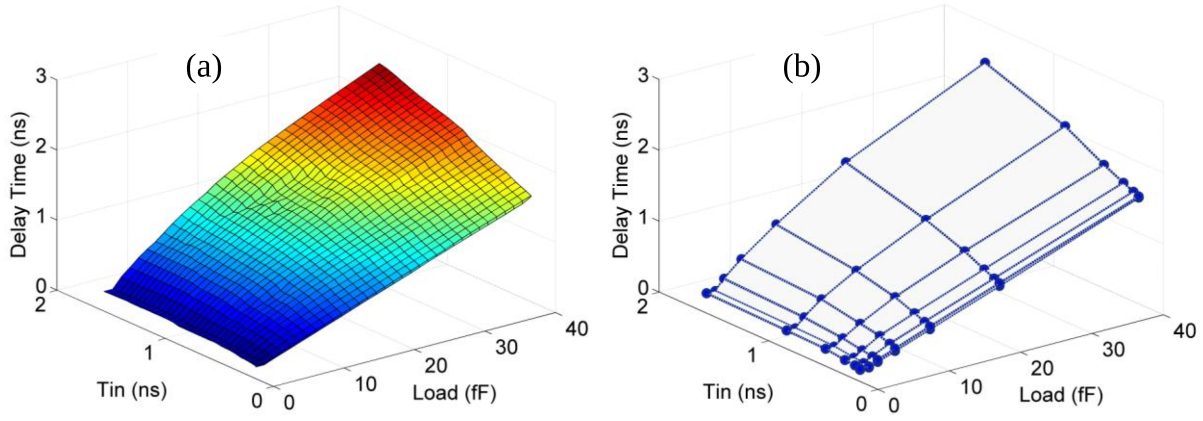


Figure 3-16 (a) Inverter delay vs. different conditions, (b) Piecewise linear fitting of inverter delay.

The sensitivity factors in the canonical delay model can be characterized in the same manner. Each variable in the canonical form needs an independent table to store its sensitivity coefficients calculated by applying the previous sampling method. Furthermore, it also needs an extra table in the delay model to store the gate output signal slopes under different operating conditions, because output signal slope will become the input signal slope for the next gate, which is an essential input parameter to the statistical cell. Figure 3-17 shows a general view of the table look-up methodology for modelling the delay variations of library cells. The rows of each page of the 3D look-up table represent the 7 typical input signal slope values, and the columns indicate the 7 typical load capacitance values of the cell. The first page of the model stores the cell mean time delay values for different operating conditions and second page stores the corresponding output signal slopes. The other pages of the model contain the cell sensitivity factors at different T_{in} and C_L values for a number of variational sources. If *n* device and environmental parameters need to be modelled, then it will be *(n+2)* pages included in the 3-D look up table shown in Figure 3-17.

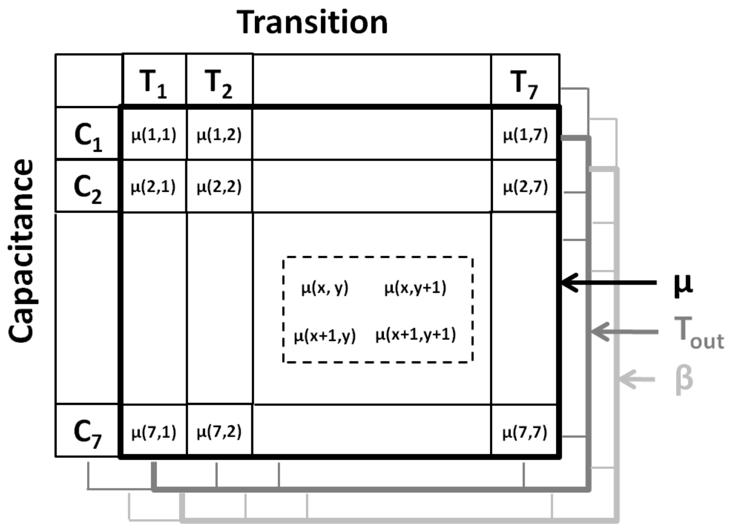


Figure 3-17 The look-up tables for modelling cell delay.

As an example, the delay model of a 2-input NAND gate cell is shown in Figure 3-18 (a) - (d). The effective gate channel length L_{eff} and supply voltage V_{dd} are selected to be the variational sources of the cell delay, whose sensitivities are demonstrated in (c) and (d) respectively. Graph (a) is for the mean delay times and (b) is for the output signal slopes of the NAND gate in different operating conditions.

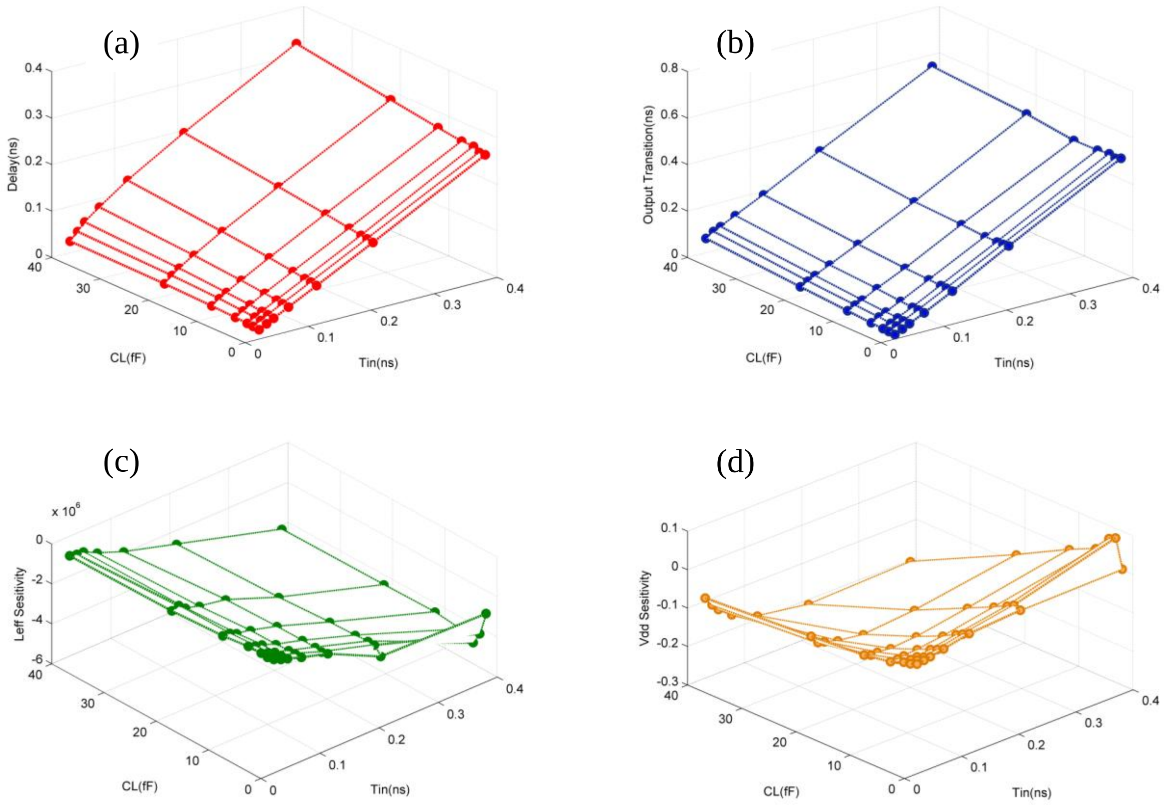
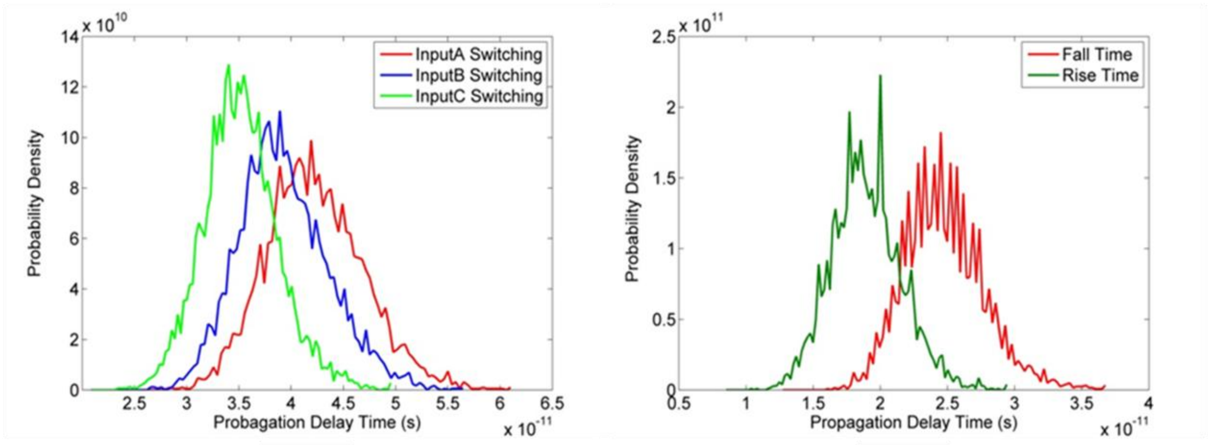


Figure 3-18 Delay model for a 2-inpu NAND gate.

On the other hand, it is necessary to distinguish the different input conditions applied to a gate in cell delay modelling, as this factor will also significantly affect its performance. The same signal transition at different gate inputs can cause varied gate delay PDF since the capacitance of each gate input is different from each other.



(a) (b)

Figure 3-19 (a) Delay PDF of a NAND gate with different input stimulus, (b) Delay PDF of a NAND gate with different output transition cases.

Consider a 3-input NAND gate with 3 different input patterns, all of which will cause the gate output to transit from a logic low to a logic high. For each input pattern, only one signal is switching, and the other two inputs stay at a logic high. The delay PDFs for the gate in each case is different from the other two, which is shown in Figure 3-19 (a). Similarly, the delay performance is also quite different during rise and fall (charge and discharge) transitions of the gate output. In Figure 3-19 (b), the rise and fall delay PDFs of the same 3-input NAND gate when only the first input signal is switching and the other two stay at a logic high, is shown again to be significantly different from each other. Consequently, it is necessary to characterize the cell for each gate input condition and transition case when modelling the library cell.

For a modelled cell in the library, it is easy to compute its delay distribution when the device variation specifications, input and load conditions are given. Algorithm 1, shown below, lists all the steps to construct a logic cell in the library of standard gate types. By using this Algorithm, all the basic cells for the logic gates in the library can be created, these include an *inverter*, 2-input *NAND NOR OR AND* gates, 3-input *NAND NOR OR AND* gates and *XOR* gate.

Algorithm 1: Statistical gate cell characterization

Input: Gate Boolean input, desired process parameters under variation and their sigma values, input signal slope and output load of gate

Output: Canonical gate delay form and gate output signal slope

```
For each gate, each input condition, each output transition case
    For each specific input signal slope and output load
        Sample the gate mean delay and output slope
    End For
    For each desired process parameter under variation
        For  each input signal slope and output load
            Calculate its sensitivity coefficient
        End for
    End for
End for
```

Figure 3-20 shows several analysis delay PDFs using the characterized NAND gate cell:

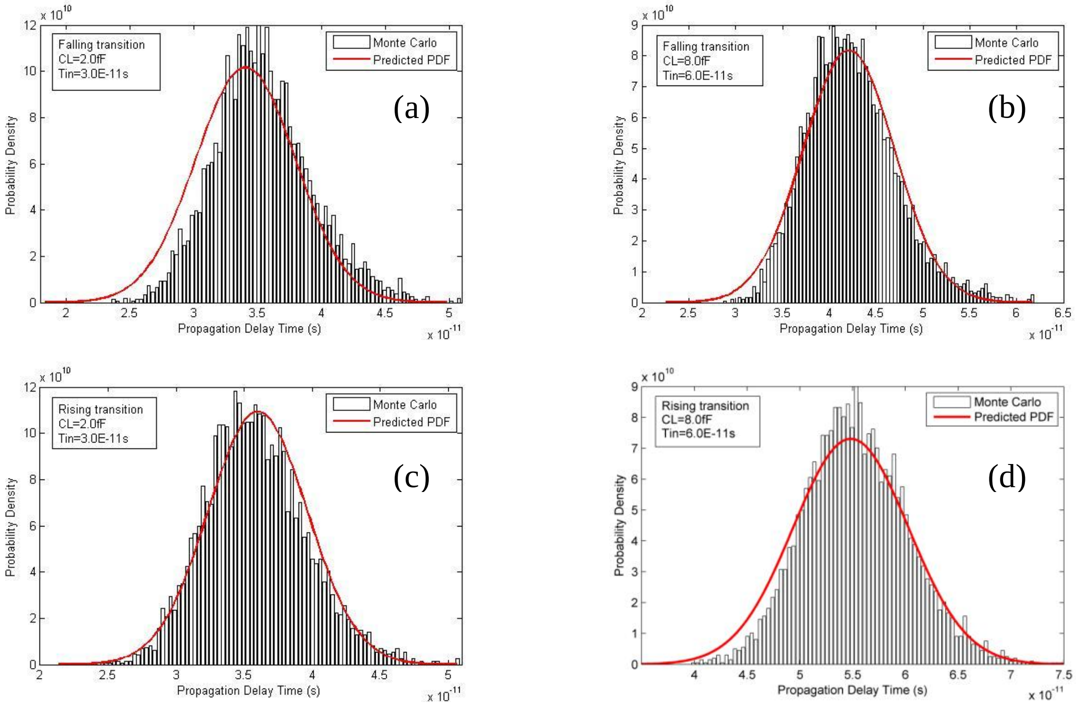


Figure 3-20 Predicted delay PDFs vs. Monte Carlo results of a NAND gate.

In the graphs above, the histograms are generated by 5000 sampled Monte Carlo simulation and the solid line is predicted using the library cell. In (a), the input signal transition time is set to 30ps and the load capacitance is set to 2fF. The operating conditions in (b) are: T_{in} = 60ps, and C_L= 8fF. Both graphs (a) and (b) are the delay distributions when the output signal of the NAND gate is a falling transition. Graphs (c) and (d) are PDF results under the same operating conditions as (a) and (b) respectively, but the output signal is a rising transition. From the PDFs matching results in Figure 3-20, it can be observed that the proposed library cell models can precisely capture the gate delay characteristics in different operating conditions and switching cases.

3.6 Summary

In this chapter, a standard cell characterization methodology has been described which takes device and environmental variation effects on circuit delay performance into account. This cell library is aimed to modeling process variation at a higher level of abstraction where the size of the circuit is large, so that lower modeling and analysis complexity is the first priority in constructing the cells. Therefore, the 1st order canonical delay model and tightness probability based SSTA technique is employed in the cell library because of their computational efficiency. The cell delay models introduced in this chapter also take different operating conditions and gate switching cases into consideration, multiple simplified look-up tables are used to capture gate delay characteristics. The approach to characterize higher level block and experimental result including the accuracy analysis will be discussed in Chapter 5. The following chapter will introduce the technique to characterize cell leakage power performance due to the process variations.

3.7 Reference

[1] D. S. Ashish Srivastava, David Blaauw *Statistical Analysis and Optimization for VLSI: Timing and Power*, 1st ed.: Springer, December 8, 2010.

[2] K. A. Bowman*, et al.*, "Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration," *Solid-State Circuits, IEEE Journal of,* vol. 37, pp. 183-190, 2002.

[3] R. B. Brawhear*, et al.*, "Predicting circuit performance using circuit-level statistical timing analysis," in *European Design and Test Conference, 1994. EDAC, The European Conference on Design Automation. ETC European Test Conference. EUROASIC, The European Event in ASIC Design, Proceedings.*, 1994, pp. 332-337.

[4] A. Gattiker*, et al.*, "Timing yield estimation from static timing analysis," in *Quality Electronic Design, 2001 International Symposium on*, 2001, pp. 437-442.

[5] A. Agarwal*, et al.*, "Statistical delay computation considering spatial correlations," in *Design Automation Conference, 2003. Proceedings of the ASP-DAC 2003. Asia and South Pacific*, 2003, pp. 271-276.

[6] L. Rung-Bin and W. Meng-Chiou, "A new statistical approach to timing analysis of VLSI circuits," in *VLSI Design, 1998. Proceedings., 1998 Eleventh International Conference on*, 1998, pp. 507-513.

[7] B. Choi and D. M. H. Walker, "Timing analysis of combinational circuits including capacitive coupling and statistical process variation," in *VLSI Test Symposium, 2000. Proceedings. 18th IEEE*, 2000, pp. 49-54.

[8] M. Orshansky and K. Keutzer, "A general probabilistic framework for worst case timing analysis," in *Design Automation Conference, 2002. Proceedings. 39th*, 2002, pp. 556-561.

[9] H. Mangassarian and M. Anis, "On statistical timing analysis with inter- and intra-die variations," in *Design, Automation and Test in Europe, 2005. Proceedings*, 2005, pp. 132-137 Vol. 1.

[10] D. Blaauw*, et al.*, "Statistical Timing Analysis: From Basic Principles to State of the Art," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 27, pp. 589-607, 2008.

[11] L. Jing-Jia*, et al.*, "Fast statistical timing analysis by probabilistic event propagation," in *Design Automation Conference, 2001. Proceedings*, 2001, pp. 661-666.

[12] L. Jing-Jia*, et al.*, "False-path-aware statistical timing analysis and efficient path selection for delay testing and timing validation," in *Design Automation Conference, 2002. Proceedings. 39th*, 2002, pp. 566-569.

[13] S. R. Naidu, "Timing yield calculation using an impulse-train approach," in *Design Automation Conference, 2002. Proceedings of ASP-DAC 2002. 7th Asia and South Pacific and the 15th International Conference on VLSI Design. Proceedings.*, 2002, pp. 219-224.

[14] S. B. McGrayne, *The Theory That Would Not Die: How Bayes' Rule Cracked the Enigma Code, Hunted Down Russian Submarines, and Emerged Triumphant from Two Centuries of Controversy*, Unabridged ed.: Tantor Media, Inc, 2012.

[15] A. Agarwal, *et al.*, "Statistical timing analysis using bounds and selective enumeration," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 22, pp. 1243-1260, 2003.

[16] A. Devgan and C. Kashyap, "Block-based static timing analysis with uncertainty," in *Computer Aided Design, 2003. ICCAD-2003. International Conference on*, 2003, pp. 607-614.

[17] A. Agarwal, *et al.*, "Statistical timing analysis for intra-die process variations with spatial correlations," in *Computer Aided Design, 2003. ICCAD-2003. International Conference on*, 2003, pp. 900-907.

[18] C. Hongliang and S. S. Sapatnekar, "Statistical timing analysis considering spatial correlations using a single PERT-like traversal," in *Computer Aided Design, 2003. ICCAD-2003. International Conference on*, 2003, pp. 621-625.

[19] L. Jiayong, *et al.*, "STAC: statistical timing analysis with correlation," in *Design Automation Conference, 2004. Proceedings. 41st*, 2004, pp. 343-348.

[20] C. Visweswariah, *et al.*, "First-Order Incremental Block-Based Statistical Timing Analysis," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 25, pp. 2170-2180, 2006.

[21] Z. Lizheng, *et al.*, "Block based statistical timing analysis with extended canonical timing model," in *Design Automation Conference, 2005. Proceedings of the ASP-DAC 2005. Asia and South Pacific*, 2005, pp. 250-253 Vol. 1.

[22] Z. Lizheng, *et al.*, "Statistical timing analysis with extended pseudo-canonical timing model," in *Design, Automation and Test in Europe, 2005. Proceedings*, 2005, pp. 952-957 Vol. 2.

[23] Z. Lizheng, *et al.*, "Correlation-preserved non-Gaussian statistical timing analysis with quadratic timing model," in *Design Automation Conference, 2005. Proceedings. 42nd*, 2005, pp. 83-88.

[24] M. Kline, *Mathematical Thought from Ancient to Modern Times*, 1st ed. vol. 1: Oxford University Press, 1990.

[25] S. Zhou, *et al.*, *Probability and Mathematical Statistics*, 4 ed.: Higher Education Press, 2008.

[26] Y. Zhan, *et al.*, "Correlation-aware statistical timing analysis with non-Gaussian delay distributions," in *Design Automation Conference, 2005. Proceedings. 42nd*, 2005, pp. 77-82.

[27] J. Watts, "Modeling circuit variability," in *Physics of Semiconductor Devices, 2007. IWPSD 2007. International Workshop on*, 2007, pp. 57-61.

[28] *International Technology Road Map for Semiconductors (ITRS)*. Available: http://public.itrs.net

[29] J. A. G. Jess, *et al.*, "Statistical Timing for Parametric Yield Prediction of Digital Integrated Circuits," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* vol. 25, pp. 2376-2392, 2006.

[30] C.E.Clark, "The greatest of a finite set of random variables," *Oper.Res,* vol. 9, pp. 145-162, 1961.

[31] M.Cain, "The moment-generating function of the minimum of bivariate normal random variables," *Amer. Star.,* vol. 48, pp. 124-125, 1994.

[32] M. Alioto, *et al.*, "Understanding the Effect of Process Variations on the Delay of Static and Domino Logic," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on,* vol. 18, pp. 697-710, 2010.

[33] A. Massimo, *et al.*, "Analysis of the impact of random process variations in CMOS tapered buffers," in *Electronics, Circuits, and Systems, 2009. ICECS 2009. 16th IEEE International Conference on,* 2009, pp. 57-60.

CHAPTER 4

CELL CHARACTERIZATION FOR LEAKAGE POWER

4.1 Introduction

Chapter 3 outlined the methodology for modelling the process variation effects on gate delay. In nanometer technology, the uncertainty in another important circuit performance parameter, namely power dissipation, due to process variability is also becoming a major issue. There are two main components of power dissipation, dynamic and static power.

Dynamic power, which is also known as switching power, corresponds to power dissipated during the signal transition of nodes in a circuit and is spent in charging capacitances associated with the transistors and wires [1]. However, the modelling of dynamic power dissipation becomes more complicated when spurious transitions or glitches are taken into consideration [2], these are the unnecessary signal transitions caused by input signals switching. There is still not an efficient model available in the literature which leads to an hierarchical design style. Most of the research into dynamic power estimation is focused on the fixed-delay model [3-5], however this model is invalid when considering the effects of process variation. Some research has been undertaken to model process variation effects on dynamic power [6, 7], but is limited to very simple models, in which only the mean values of the power dissipation are considered. Therefore, there are still major difficulties for characterizing dynamic power dissipation at a higher level of design abstraction.

On the other hand, the static power, which is also known as leakage power, has grown significantly with the drastic scaling of semiconductor technology and contributes a huge fraction of the total power budget. A study from the Intel Corporation shows that leakage power will contribute approximately 50% of the total power dissipation at the 90nm technology node [1], and the percentage will grow larger in more advanced technologies. Thus the effect of process variability on such an important and variability-sensitive parameter needs more attention. Therefore, the power dissipation modelling work in this thesis is only focused on the leakage power.

The organization of this chapter is as follow: a brief overview of leakage current mechanism will be outlined in Section 4.2, including its main components and causes. In Section 4.3 and 4.4, the leakage power modelling and analysis techniques which will be employed in the cell library will be described with discussions on why they are selected. The methodology to characterize leakage power performance of a library cell will be introduced in Section 4.5 and summary will be outlined at the end of this chapter.

4.2 Overview of Leakage Power

The leakage power or static power is refers to the unwanted energy dissipation when the electronic device is in an off or standby mode. It is caused by the leakage current flow which should be zero ideally. There are a number of phenomena which contribute to the generation of the device leakage current I_{off}, which is the reason for the unexpected static power dissipation. Eight different leakage current mechanisms have been listed in [8]. However, not all of these components of leakage current are significant. The main contributors to the device leakage current are the subthreshold leakage (I_{sub}) and gate leakage current (I_{gate}).

The subthreshold leakage, which is also known as subthreshold conduction or subthreshold drain current, is the current that flows between the source and drain of a MOSFET when the transistor is in the subthreshold region, or weak-inversion region, that is when the gate-to-source voltage falls below the threshold voltage. Figure 4-1 (a) illustrates the subthreshold leakage current flow in an n-type MOSFET. The reason for the growing importance of subthreshold leakage is that the supply voltage is continued to be scaled down in order to keep the electrical field inside smaller devices low and thus maintain their reliability. Consequently there is less gate voltage swing below threshold voltage to turn the device off. Since I_{sub} varies exponentially with gate voltage, it becomes more and more significant as MOSFETs shrink in size.

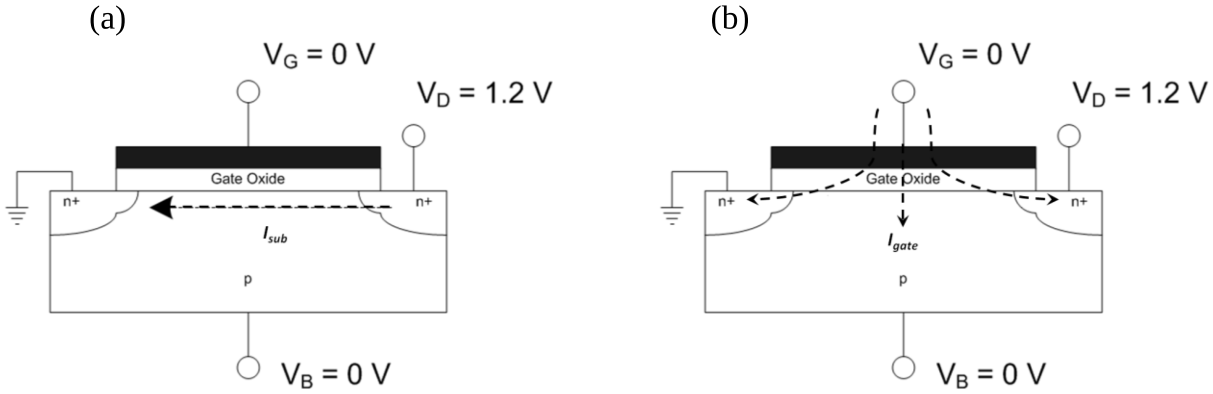


Figure 4-1 Leakage current mechanisms: (a) subthreshold leakage, (b) gate leakage.

Another component of the gate leakage current results from the tunneling of electrons (holes) from the substrate to the gate of a NMOS (PMOS) device through the gate oxide layer as shown in Figure 4-1 (b). The gate oxide serves as an insulator between the gate and the channel of devices and ideally can block the any current flow. However, in order to satisfy the aggressive scaling of semiconductor technology to respond to the market demand for better device performance, the oxide layer should be made as thin as possible. Unfortunately, the possibility of electron tunneling occurring will increase with the continued shrinking of the oxide thickness T_{ox}, which leads to a larger leakage power dissipation. Since the gate leakage current has an exponential relationship with T_{ox}, which is one the most sensitive device parameters under process variation effects, the resulting leakage current distribution is also quite significant and cannot be neglected.

Additionally, due to the exponential relationship between I_{off} and V_{th}, the leakage current of a device not only grows rapidly but also shows large fluctuations from die to die and even from gate to gate. This is especially true in nanometre technology where controlling V_{th} is extremely difficult because of the drain-induced barrier lowering effects (DIBL) [9]. DIBL has become a serious problem which limits the MOSFET performance since the device channel length first reached submicron dimensions, and it is exacerbated in sub-100nm devices by fundamental scaling limitation on oxide thickness [10].

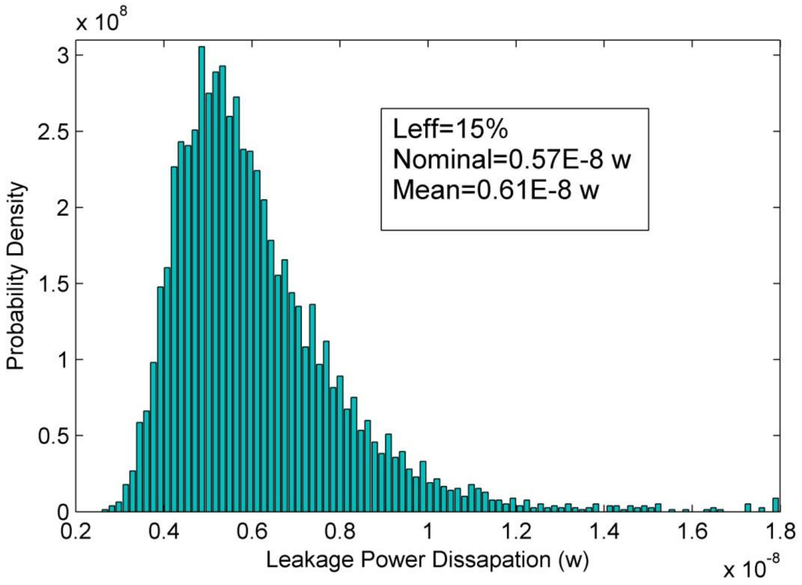


Figure 4-2 Leakage power distribution of an inverter.

The estimation of leakage power distributions becomes difficult with the growing uncertainty in leakage current due to the variational device parameters, such as threshold voltage and gate oxide thickness. The traditional worse-case and corner-based approaches become impractical for the leakage power analysis, because the leakage power has a very wide distribution. Figure 4-2 shows the static power PDF of an inverter circuit with the device gate length L_{eff} deviating ±15% from its nominal value, where L_{eff} is assumed to be a Gaussian variable. It can be seen that the power leakage of the circuit could be almost 4 times larger than its nominal value. Additionally, the nominal values of process variation parameters do not correspond to the average value of leakage power, since, as shown in Figure 4-2, the distribution is no longer normal with respect to the Gaussian source variable. Such characteristics of the distribution become crucial in the analysis of circuit leakage power dissipations.

The worse-case model files for leakage current can easily exhibit 10-100 times larger I_{off} than a nominal device [11]. This will lead to an overly conservative analysis result and unnecessarily raise the power specification of a circuit design. On the other hand, the leakage power variation also cannot be ignored. A small number of very leaky devices can easily dominate the static power consumption in a circuit block. Figure 4-3 shows that the average leakage current can be much larger (~30% for PMOS with L $3\sigma/\mu$=12.5%) than the nominal leakage due to the exponential dependence of current on the gate length

[10]. The results also shows that the gate length variation effects on PMOS are much greater than on NMOS, this is because DIBL effects in PMOS devices are typically more significant than in NMOS devices [12].

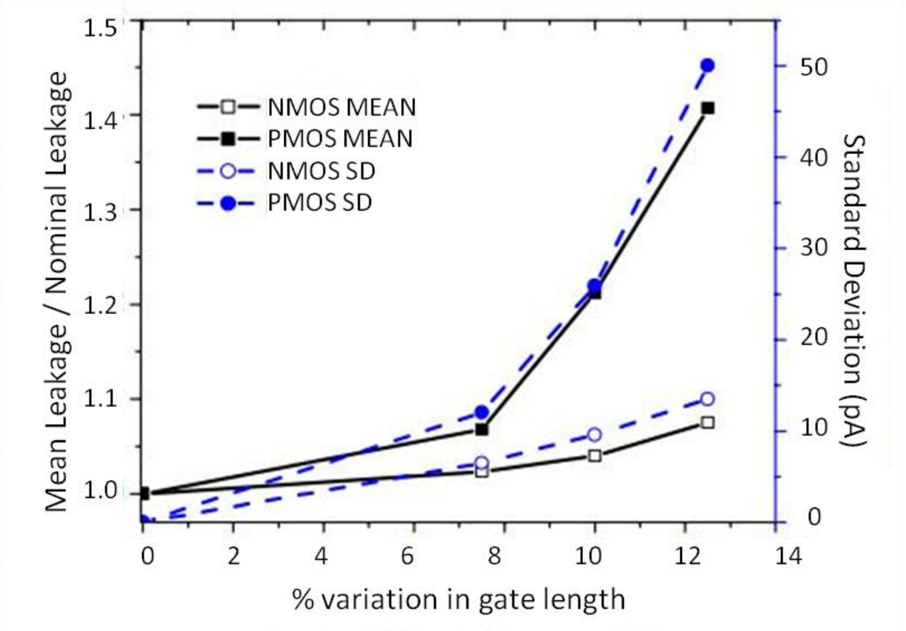


Figure 4-3 Dependence of mean and standard deviation of leakage current on 3σ variation in gate-length [10].

Based on the discussion in this section, the leakage power has becomes a major problem due to the aggressive increase in device leakage current. The traditional worst-case model fails to cope with the widely distributed I_{off} and provide a reasonable analysis result. Monte Carlo techniques can precisely predict the leakage power performance of a circuit but are very expensive in terms of computational time and complexity. Thus an efficient static power analysis technique for circuits is needed so that power dissipation can be estimated before circuit are fabricated. The following sections will discuss the leakage power modelling and analysis methodologies in detail.

4.3 Leakage Current and Power Models

In this section, the analytical leakage current models will first be introduced, using the equations to model the subthreshold current and gate leakage current. Followed by a description of the statistical leakage power modelling technique, in which both inter-die and intra-die variations are considered. An explanation will then be given to justify why the latter approach is more appropriate for building up the cell library.

4.3.1 Analytical leakage models

The starting point for traditional models for static power analysis are the analytical equations for computing leakage current I_{off}. As discussed in the previous section, there are 2 main components in I_{off}; the subthreshold current and gate leakage current. The subthreshold current is the current that flows between the source and drain of a device when the device is turned off, and it can be expressed as Equations 4.1 and 4.2 [8]:

$$I_{sub} = I_0\, exp\left(\frac{V_{gs} - V_{th}}{nV_T}\right)\left(1 - exp\left(\frac{-V_{ds}}{V_T}\right)\right) \tag{4.1}$$

where

$$I_0 = \mu_0\, C_{ox}\left(\frac{W}{L_{eff}}\right) V_T^2 (n-1) \tag{4.2}$$

and μ_0 is the charge-carrier effective mobility, n is the subthreshold slope factor and C_{ox} is the gate oxide capacitance. V_T=KT/q is the thermal voltage, where K is the Boltzmann constant, T is the absolute temperature and q is the electron charge. V_{th} is the device threshold voltage which can be expressed as Equation 4.3 [8]:

$$V_{th} = V_{fb} + |2\phi_p| + \frac{\lambda_b}{C_{ox}}\sqrt{2qN_{ch}\epsilon_s(|2\phi_p| + V_{sb})} - \lambda_d V_{ds} \tag{4.3}$$

Where V_{fb} is the flat-band voltage, ϕ_p is the surface potential, λ_b is the body effect factor, N_{ch} is the channel doping concentration, ϵ_s is the permittivity of silicon and λ_d is the DIBL coefficient.

On the other hand, the gate leakage current, which is caused by electron tunnelling phenomenon, is composed of several components as shown in Figure 4-4. I_{gos} and I_{god} are the leakage current flow through the gate-to-source/drain extension overlap regions, I_{gcs}

and I_{gcd} are the leakage currents between the gate and source/drain diffusion through the channel region, and I_{gb} is the leakage current between gate and body.

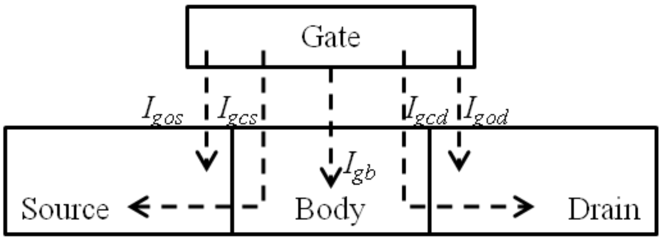


Figure 4-4 Components of gate tunnelling current

The key dependency of gate leakage current on the process parameters can be expressed as Equation 4.4 [8], where A_g and B_g are process dependent physical parameters. This equation shows that the gate leakage current is an exponential function of the gate oxide thickness.

$$I_{gate} = WA_g \left(\frac{V_{dd}}{T_{ox}}\right)^2 exp\left(-B_g \frac{T_{ox}}{V_{dd}}\right) \tag{4.4}$$

Most of the analytical approaches to estimate circuit static power dissipation are based on the above leakage current models from Equation 4.1 to 4.4 [10, 13, 14]. However, these approaches are faced with some drawbacks. Firstly, the analytical equations introduced in this subsection are the simplified expressions, the actual BSIM model used to compute leakage current in SPICE is much more complicated. It is shown in [10] that the above equations are not very accurate for 0.18μm technology. Therefore the predicted result for leakage current based on the analytical models is very suspect. On the other hand, the computational complexity of the analytical models in Equations 4.1 to 4.4 is still quite high when considering the analysis of leakage power in a large circuit which contains thousands of transistors, even though these are already in simplified forms. Therefore, a much more efficient and accurate leakage power model is needed for characterizing library cells.

4.3.2 Statistical gate leakage power models

Similar to the canonical gate delay model, the gate leakage power dissipations can also be modelled as low-order polynomials in a statistical manner. All the variational sources are assumed to be Gaussian variables. Since the leakage current has an exponential relationship to the device parameters, the distribution of the leakage power is in a close to lognormal form. The 1st order gate leakage power model is shown in Equation 4.5 [15]:

$$LP(Leakage\ Power) = exp\left(\mu_P + \sum_{i=1}^{n} \beta_{Pi} G_i + \beta_{P(n+1)} R\right) \tag{4.5}$$

The terms inside "*exp()*" is exactly the same form as the 1st order canonical delay model. The dependency of gate leakage power on a process parameter can be simply represented by a sensitivity factor, which saves a huge amount of computational time. On the other hand, the values of $\beta_p s$ are computed by finite different approach based on the SPICE simulation results which are much more accurate than the analytical models. Therefore, the estimated distribution of gate leakage power using the canonical model is more reliable. The expression for β_p of the corresponding variable *x* is shown Equation 4.6:

$$\beta = \frac{ln\big(f(\mu_x + \sigma_x)\big) - ln\big(f(\mu_x - \sigma_x)\big)}{2\sigma_x} \tag{4.6}$$

where μ_x and σ_x are the mean value and standard deviation of the variable *x*, *f* represents the SPICE response of the circuit static power for a given device parameter value.

In order to demonstrate the accuracy of the canonical model, Figure 4-5 shows the Monte Carlo simulation result of an inverter leakage power distribution and the corresponding lognormal fitting plot with the ITRS 90nm technology variation specifications, which are indicated inside the graph. The two plots in the figure are well matched, which illustrates that the distribution of gate leakage power dissipation can be approximated as a lognormal model using Equation 4.5 with reasonable accuracy.

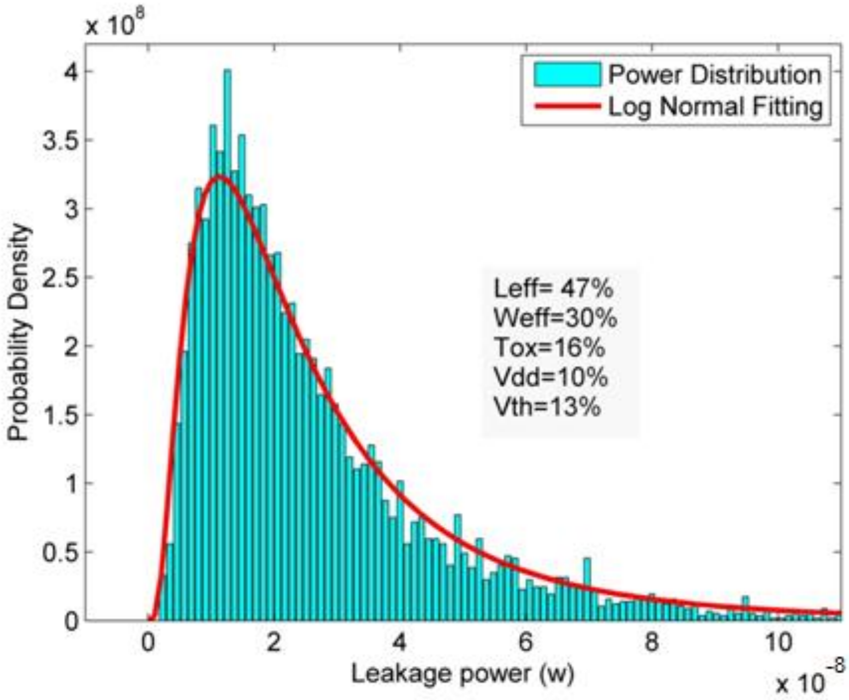


Figure 4-5 Inverter leakage power PDFs of 90nm technology.

Due to the great trade off between the modelling accuracy and computational complexity, the 1st order canonical gate leakage power model will be used for characterising the library cell to analyze the process variation effects on circuit leakage power. In the following section, the analysis approach to estimate leakage power PDFs when a circuit comprises multiple gates will be discussed in detail.

4.4 Statistical Analysis for Leakage Power

Computing the total leakage power dissipation of a circuit which contains multiple gates comprises, simply, adding the individual gate leakage power values together. Since the gate leakage power model moves from the deterministic domain to the statistical domain in order to take process variation effects into account, the total leakage power of a circuit can be expressed as the sum of gate model variables represented by a probability distribution. As described in the previous section, the cell leakage power can be modelled as a lognormal variable, then the total static power can be expressed as Equation 4.7:

$$S = X_1 + X_2 + \cdots + X_n = e^{Y_1} + e^{Y_2} + \cdots + e^{Y_n} \tag{4.7}$$

where X_i represent the i^{th} independent lognormal gate model in a circuit. The leakage power analysis does not need the non-linear statistical maximum operation as in SSTA. However, the variable addition is not straightforward since the variables are in a non-linear lognormal form. Theoretically, the sum of multiple lognormal distributed variables has no close-form expression. Various approaches have been proposed to estimate the sum result of multiple lognormal variables. A full comparison of a number of lognormal summation approaches has been undertaken [16], the conclusion is that the simple *Wilkinson's approximation* [17] is more accurate than other complex techniques for computing the leakage power PDFs based on matching the first two moments. In *Wilkinson's approximation,* the mean value and standard deviation of the sum of *n* independent lognormal gate leakage power models can be expressed as Equation 4.8 and Equation 4.9 respectively:

$$\mu[S] = E[S] = \mu_1 + \mu_2 + \cdots + \mu_n \tag{4.8}$$

$$\sigma[S] = \sqrt{\sigma_1^2 + \sigma_2^2 + \cdots + \sigma_n^2} \tag{4.9}$$

where μ_i and σ_i represents the mean value and standard deviation of i^{th} individual gate leakage power model. Each lognormal variable can be expressed as Equation 4.10:

$$f(x) = \frac{1}{x\beta\sqrt{2\pi}} exp\left(\frac{-(\ln(x) - \alpha)^2}{2\beta^2}\right) \tag{4.10}$$

where α and β are the parameters which define the shape of a lognormal PDF. If *Y(u, σ)* is a Gaussian variable and its corresponding lognormal form is expressed as *X=exp(Y),* then the parameters α and β in *X* are the mean value and standard deviation of *Y*. These parameters can be used to compute the mean value and variance of the lognormal variable *X* as Equation 4.11 and 4.12: (Note that the mean and sigma value of the Gaussian variable *Y* are not the same as in the corresponding lognormal variable *Y*)

$$E[X] = exp(\alpha + \beta^2/2) \tag{4.11}$$

$$Var[S] = exp(2\alpha + 2\beta^2) - exp(2\alpha + \beta^2) \tag{4.12}$$

Similarly, the mean value and variance of X can also be used to compute the mean and sigma values (α and β) of Y using Equations 4.13 and 4.14.

$$\alpha = \frac{1}{2}\, ln\left(\frac{E[X]^4}{E[X]^2 + Var[X]}\right) \tag{4.13}$$

$$\beta^2 = \, ln\left(\frac{Var[X] + E[X]^2}{E[X]^2}\right) \tag{4.14}$$

Based on the equations above, the lognormal parameters of the S, which is the sum of multiple lognormal variables shown in Equations 4.8 and 4.9, can be obtained permitting S to be re-expressed into lognormal form. It is interesting to note that if the value of n is large which indicates the size of the circuit under analysis, the leakage power PDF will approach a Gaussian distribution theoretically due to the *central limit theorem* [18]. This characteristic is also true in real circuit power analysis..

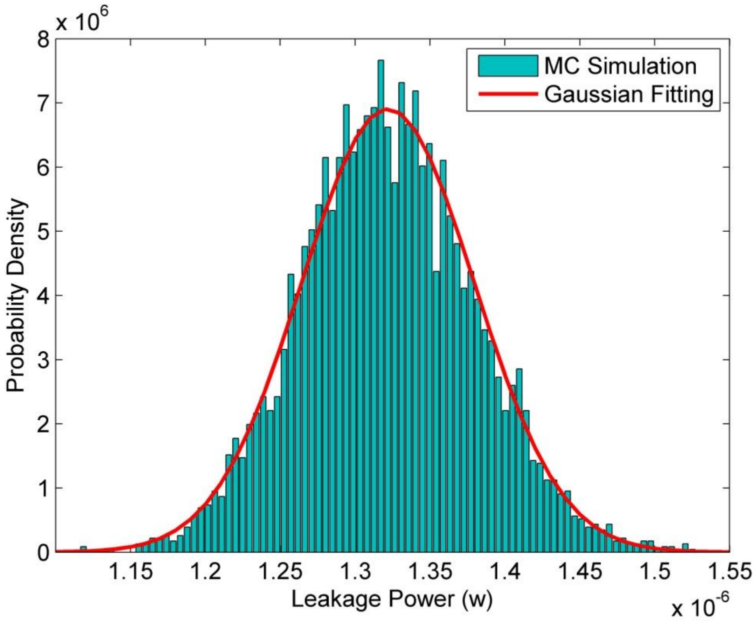


Figure 4-6 PDFs of a 200-gate inverter chain circuit.

Figure 4-6 shows the Monte Carlo simulation result (histogram) and its normal fitting graph (solid line) for the total static power distribution of a 200-gate inverter chain. All the inverters in the circuit are identical. The total distribution in the graph shows a high degree of normality and is quite close to a Gaussian distribution even though each individual inverter leakage power PDF is lognormally distributed which has already been shown in Figure 4-2 and Figure 4-5.

The *Wilkinson's approximation* can provide good modelling accuracy for computing the sum of the gate leakage power models which are independent (intra-die). The inter-die variations are totally ignored. In [19], the author proposed a expanded approach to power analysis based on *Wilkinson's approximation*, which uses the 1st order canonical form as the gate leakage power model so that both inter- and intra-die variation effects are considered. However, in this technique, the RVs associated with all the cells in a circuit must be summed in a single step and the summed result cannot be re-expressed back into a canonical form. Since this power analysis technique involves *n*-by-*n* matrix multiplications, where *n* is the number of the gates in a circuit, the overall complexity will be $O(n^2)$ [20]. Additionally, this approach cannot be used in an incremental design such as building up the cell library.

A second extension of *Wilkinson's approximation* [15] has been proposed as a simplified leakage power analysis approach, which uses a recursive technique to reduce the computation complexity of the summation of lognormal (canonical form) power RVs; only two RVs are summed in one step, and then the sum will be re-expressed in a canonical form for the next summation step. Therefore, the computational complexity has been reduced to $O(n)$. Assuming each gate leakage power distribution, P_x, in a circuit is modelled in the 1st order canonical form in Equation 4.5, then its mean and variance can be computed using Equations 4.15 and 4.16:

$$E[P_x] = exp\left(x_0 + \frac{1}{2}\sum_{i=1}^{n+1} x_i^2\right) \tag{4.15}$$

$$Var[P_x] = exp\left(2x_0 + 2\sum_{i=1}^{n+1} x_i^2\right) - exp\left(2x_0 + \sum_{i=1}^{n+1} x_i^2\right) \tag{4.16}$$

The correlation of the leakage power of cell *x* with the lognormal variable associated with the global component G_j in the canonical leakage power model,   as shown in Equation 4.5, is computed using Equation 4.17.

$$E\left[P_x e^{G_j}\right] = exp\left(\mu_x + \sum_{i=1,i\neq j}^{n} \beta_{xi}^2 + \left(\beta_{xj}+1\right)^2\right),\ \ \forall j \in \{1,2,\dots,n\} \tag{4.17}$$

Assuming P_a is the sum of the leakage power of two cells b and c, it needs to be expressed into the canonical form as in Equation 4.18 for further calculations using leakage power analysis.

$$P_a = P_b + P_c = exp\left(\mu_a + \sum_{i=1}^{n} \beta_{ai}G_i + \beta_{a(n+1)}R\right) \tag{4.18}$$

The covariance of the leakage power of b and c can be obtained from Equations 4.19 and 4.20 below:

$$Cov(P_b, P_c\ ) = E[P_b \cdot P_c\ ] - E[P_b] \cdot E[P_c\ ] \tag{4.19}$$

$$E[P_b \cdot P_c\ ] = exp\left((\mu_b + \mu_c) + \frac{1}{2}\left(\sum_{i=1}^{n}(\beta_{bi} + \beta_{ci})^2 + \beta_{b(n+1)}^2 + \beta_{c(n+1)}^2\right)\right) \tag{4.20}$$

Finally, all the coefficients of P_a can be computed by using Equations 4.21 – 4.22:

$$\beta_{ai} = log\left(\frac{E[P_b e^{G_i}] + E[P_c e^{G_i}]}{(E[P_b] + E[P_c])E[e^{G_i}]}\right) \tag{4.21}$$

$$\mu_a = \frac{1}{2}log\left(\frac{(E[P_b] + E[P_c])^4}{(E[P_b] + E[P_c])^2 + Var(P_b) + Var(P_c) + 2Cov(P_b, P_c\ )}\right) \tag{4.22}$$

$$\beta_{a(n+1)} = \sqrt{\log\left(1 + \frac{Var(P_b) + Var(P_c) + 2Cov(P_b, P_c\ )}{(E[P_b] + E[P_c])^2}\right) - \sum_{i=1}^{n} {\beta_{ai}}^2} \tag{4.23}$$

Figure 4-7 shows the flowchart for computing the total leakage power dissipation of a circuit. For a given netlist, the canonical leakage power models for all the gates will be identified and placed into a model vector, ready to be processed. Only two of models are summed at a time using Equations 4.15 to 4.23. The expected values and variances of A and B (A and B could be any two models in the model vector) will be computed first, these values can be used to further calculate the correlation and covariance of A and B. Subsequently the two lognormal models can be added using the previously obtained interim results. The sum of A and B is still in canonical form and will be placed back into the model vector. The whole calculation process continues until there is only one model remaining in the model vector, which is the expression of the total circuit leakage power distribution.

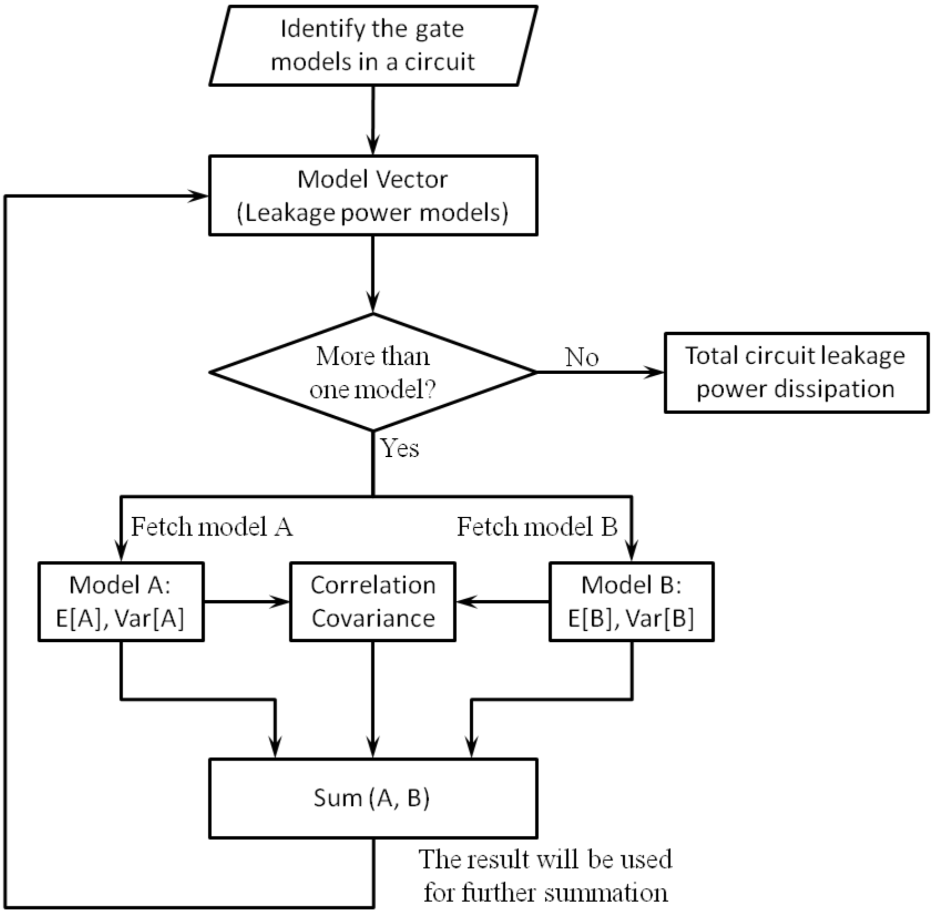

Identify the gate models in a circuit
Model Vector (Leakage power models)
More than one model?
No
Total circuit leakage power dissipation
Yes
Fetch model A
Fetch model B
Model A: E[A], Var[A]
Correlation Covariance
Model B: E[B], Var[B]
Sum (A, B)
The result will be used for further summation

Figure 4-7 Flowchart for computing the total leakage power of a circuit.

Based on this simplified power analysis technique, the distribution of the total leakage power dissipation for a multiple-gate circuit can be estimated efficiently. Based on the experimental results in [15], this approach also provides a reasonable accuracy with an average error rate of less than 5%. Most importantly, it leads to an incremental design style making it possible to be employed in the cell library characterization. Having established the approaches to model gate static power and compute the total gate leakage power distribution due to the process variation effects, the leakage power performance of the library cells can be characterized, the corresponding methodology will be introduced in the next section.

4.5 Cell Characterization for Leakage Power

When characterising the gate leakage power dissipation, P_{leak}, there is no need to consider the input signal slope because there are no signal transitions occurring in the static circuit state. Additionally, the effects of load capacitances on the gate leakage power distribution are very small and negligible.

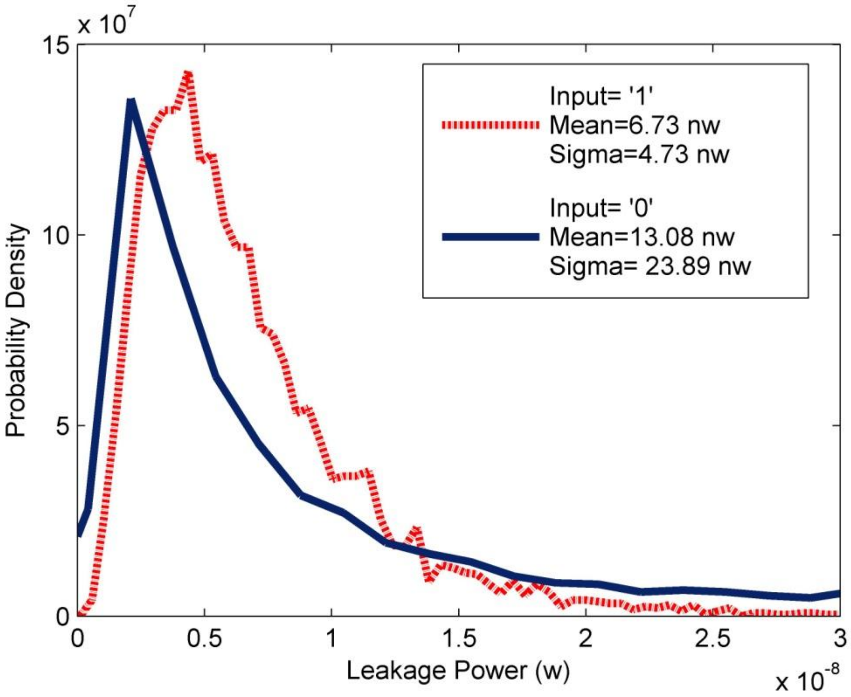


Figure 4-8 Leakage power PDFs for an inverter with different gate states.

However, as discussed in Section 4.2, the leakage current behaves differently for N and P type transistors. Furthermore, the transistor gate voltage is also a major factor which affects the cell leakage current distribution, and there could be several different gate input conditions for a cell. Therefore, leakage power distribution of a logic gate cannot be represented using a single model. P_{leak} is very dependent on the different gate static states. Figure 4-8 shows the leakage power PDFs of an inverter circuit when the input signal is at a logic low and high respectively. Significant differences in the two PDFs can be seen from the graph.

In order to capture the cell leakage power characteristics, each input state of a logic gate needs an independent model. Assuming n is the number of inputs to a gate; it will be n^2 possible states and requires n^2 canonical polynomials to model its leakage power distributions. If k variational sources need to be considered in the leakage power model, then $k+1$ values (k sensitivity coefficients for all the variation sources and 1 mean leakage power value in the lognormal canonical form) need to be stored in the memory for each canonical polynomial. Algorithm 2 shows the process to characterize the leakage power consumption for the statistical library cell.

Algorithm 2: Statistical gate leakage power characterization

Input: Gate static state, desired process parameters under variation and their sigma values

Output: Canonical polynomial of the logarithm of gate leakage power

```
For each gate, each static state
    Sample the gate leakage power and do logarithm
    For each desired process parameter under variation
        Calculate the logarithm of its sensitivity coefficient
    End for
End for
```

One look-up-table (LUT) is sufficient to model the process variation effects on cell leakage power dissipation. Figure 4-9 shows the general view of the cell leakage power model. The rows of the LUT represent different cell states; the first column is for storing the mean leakage power values and the other columns indicate all the necessary coefficients of the 1st order lognormal canonical form for different variation sources.

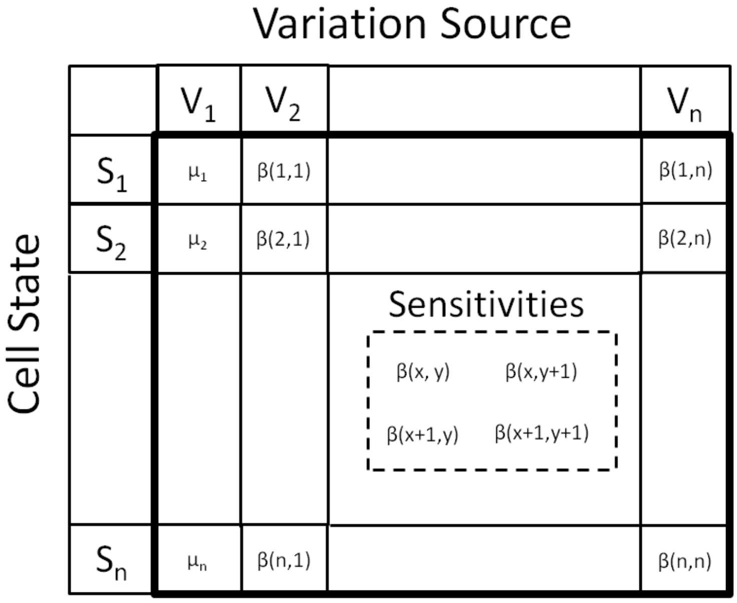


Figure 4-9 Look-up-table for modelling cell leakage power dissipation.

Figure 4-10 shows the LUT, as an example, which stores the coefficients of cell leakage power models for a 3-input NAND gate.

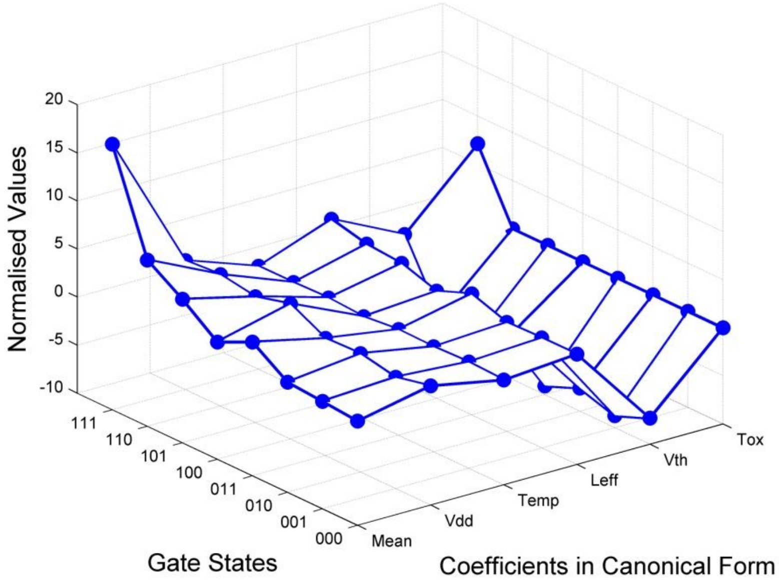


Figure 4-10 Leakage power models for a 3-input NAND gate.

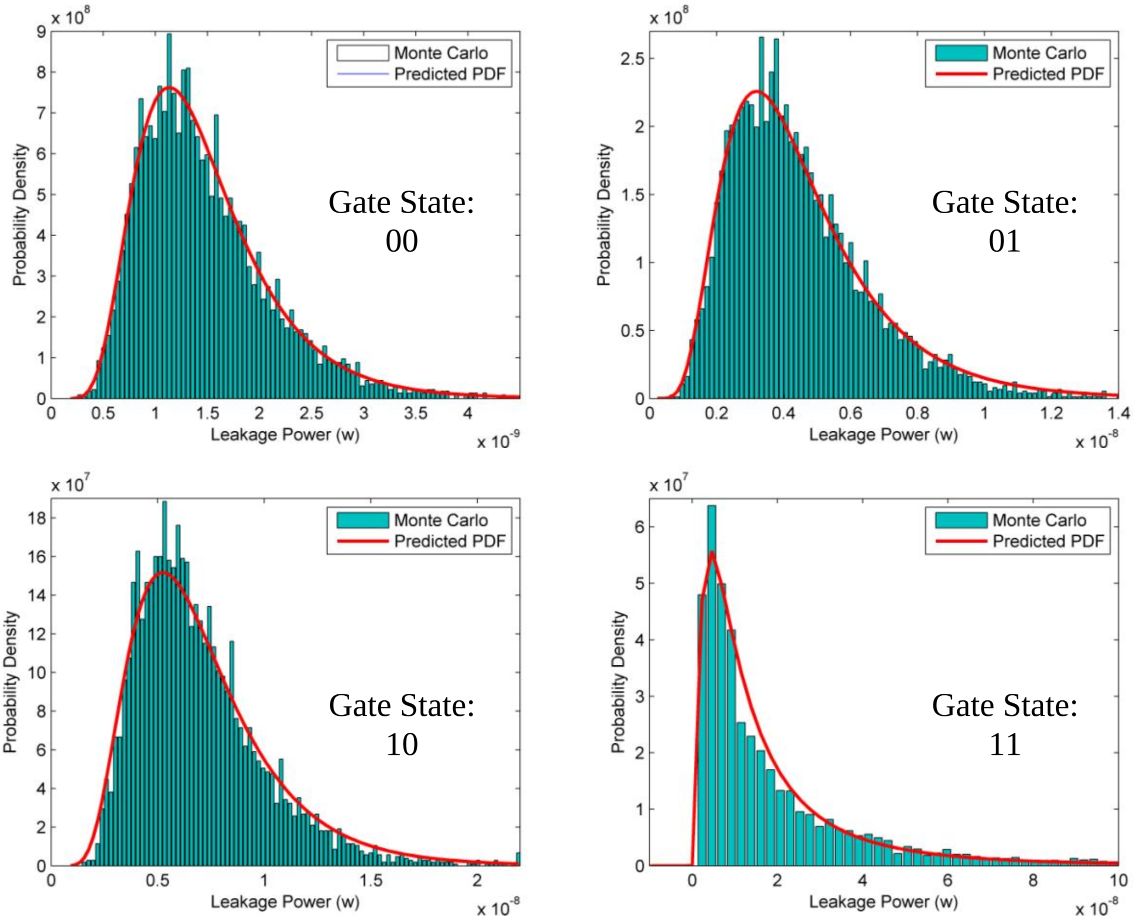


Figure 4-11 Predicted leakage power PDFs vs. Monte Carlo results of a 2-input NAND gate with different input states.

Figure 4-11 shows the leakage power PDFs of a 2-input NAND gate for the all possible 4 input states. The histogram is generated by 5000-sampled Monte Carlo simulation as the reference distribution, and the solid line is predicted values using the characterized library cell. It can be seen that these PDFs are well matched which validates accuracy of the proposed leakage power models.

4.6 Summary

A detailed description for characterising gate leakage power distribution due to the process variation effects is outlined in this chapter. The switching power is not considered in the cell library because there are still some technical difficulties in modelling this performance parameter, especially when taking glitch effects into account. Furthermore, the literature shows that the contribution of the leakage power to the total circuit power dissipation is over 50% in the 90nm technology node, and the percentage will easily go up with the continuous shrinking of the transistor dimensions. This indicates that the unwanted leakage power dissipation has already become a major problem in VLSI designs and needs to be modelled before the circuit is fabricated. Therefore, this thesis only focuses on the characterization of leakage power dissipation.

The leakage power is caused by the undesired current flow inside the devices when they are in off mode. The subthreshold current and gate leakage current are the two major component of the total device leakage current. The traditional simplified analytical models for the gate leakage current lose its accuracy when the devices become smaller and their characteristics become more complicated. Therefore, the estimation of gate leakage power distribution based on these analytical leakage current models becomes suspect. On the other hand, the statistical leakage power model, such as the lognormal canonical model, becomes more and more popular since it drives the sensitivity of each considered variation source based on the SPICE simulation results using BSIM model parameters, which are the most accurate data available Furthermore, the computational complexity of the statistical models are also much lower than the analytical methods, which makes it easier to implement in the cell library.

The total leakage power distribution of a circuit is the sum of all the individual gate static power consumption. Since the leakage power PDF of a gate is in a lognormal shape and the sum of lognormal variables has no closed form, the total static power dissipation can only be approximated. The recursive static power analysis technique based on the *Wilkinson's approximation* is employed in the proposed cell library because of its lower computational complexity compared with other approaches. All the possible input states of a gate will be considered when characterising the cell leakage power distributions, and the gate leakage power PDF in each state will be represented by an independent lognormal canonical model. Experimental results show that the proposed static power modelling approach can precisely capture the leakage characteristics of different gates.

4.7 Reference

[1] D. S. Ashish Srivastava, David Blaauw *Statistical Analysis and Optimization for VLSI: Timing and Power*, 1st ed.: Springer, December 8, 2010.

[2] D. Quang, *et al.*, "Dynamic power estimation for deep submicron circuits with process variation," in *Design Automation Conference (ASP-DAC), 2010 15th Asia and South Pacific*, 2010, pp. 587-592.

[3] C. Y. Tsui, *et al.*, "Efficient estimation of dynamic power consumption under a real delay model," in *Computer-Aided Design, 1993. ICCAD-93. Digest of Technical Papers., 1993 IEEE/ACM International Conference on*, 1993, pp. 224-228.

[4] G. Theodoridis, *et al.*, "An efficient probabilistic method for logic circuits using real delay gate model," in *Circuits and Systems, 1999. ISCAS '99. Proceedings of the 1999 IEEE International Symposium on*, 1999, pp. 286-289 vol.1.

[5] D. Sinha, *et al.*, "A Timing Dependent Power Estimation Framework Considering Coupling," in *Computer-Aided Design, 2006. ICCAD '06. IEEE/ACM International Conference on*, 2006, pp. 401-407.

[6] S. Pilli and S. S. Sapatnekar, "Power estimation considering statistical IC parametric variations," in *Circuits and Systems, 1997. ISCAS '97., Proceedings of 1997 IEEE International Symposium on*, 1997, pp. 1524-1527 vol.3.

[7] J. D. Alexander and V. D. Agrawal, "Algorithms for Estimating Number of Glitches and Dynamic Power in CMOS Circuits with Delay Variations," in *VLSI, 2009. ISVLSI '09. IEEE Computer Society Annual Symposium on*, 2009, pp. 127-132.

[8] A. Chandrakasan, *et al.*, *Design of High-Performance Microprocessor Circuits*, 1 ed.: Wiley-IEEE Press, 2000.

[9] A. Asenov, "Random dopant induced threshold voltage lowering and fluctuations in sub-0.1 &mu;m MOSFET's: A 3-D &ldquo;atomistic&rdquo; simulation study," *Electron Devices, IEEE Transactions on*, vol. 45, pp. 2505-2513, 1998.

[10] R. Rao, *et al.*, "Statistical estimation of leakage current considering inter- and intra-die process variation," in *Low Power Electronics and Design, 2003. ISLPED '03. Proceedings of the 2003 International Symposium on*, 2003, pp. 84-89.

[11] S. Narendra, *et al.*, "Full-chip sub-threshold leakage power prediction model for sub-0.18 &mu;m CMOS," in *Low Power Electronics and Design, 2002. ISLPED '02. Proceedings of the 2002 International Symposium on*, 2002, pp. 19-23.

[12] S. Tyagi, *et al.*, "A 130 nm generation logic technology featuring 70 nm transistors, dual Vt transistors and 6 layers of Cu interconnects," in *Electron Devices Meeting, 2000. IEDM '00. Technical Digest. International*, 2000, pp. 567-570.

[13] J. Kao, *et al.*, "Subthreshold leakage modeling and reduction techniques [IC CAD tools]," in *Computer Aided Design, 2002. ICCAD 2002. IEEE/ACM International Conference on*, 2002, pp. 141-148.

[14] A. Srivastava, *et al.*, "Modeling and analysis of leakage power considering within-die process variations," in *Low Power Electronics and Design, 2002. ISLPED '02. Proceedings of the 2002 International Symposium on*, 2002, pp. 64-67.

[15] A. Srivastava, *et al.*, "A Novel Approach to Perform Gate-Level Yield Analysis and Optimization Considering Correlated Variations in Power and Performance," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* vol. 27, pp. 272-285, 2008.

[16] N. C. Beaulieu, *et al.*, "Comparison of methods of computing lognormal sum distributions and outages for digital wireless applications," in *Communications, 1994. ICC '94, SUPERCOMM/ICC '94, Conference Record, 'Serving Humanity Through Communications.' IEEE International Conference on*, 1994, pp. 1270-1275 vol.3.

[17] S.C.Schwartz and Y.S.Yeh, "On the distribution function and moments of power sums with lognormal components," *Bell Syst. Tech. J.,* vol. 61, pp. 1441-1462, 1982.

[18] S. Zhou, *et al.*, *Probability and Mathematical Statistics*, 4 ed.: Higher Education Press, 2008.

[19] A. A. Abu-Dayya and N. C. Beaulieu, "Outage probabilities in the presence of correlated lognormal interferers," *Vehicular Technology, IEEE Transactions on,* vol. 43, pp. 164-173, 1994.

[20] C. Hongliang and S. S. Sapatnekar, "Full-chip analysis of leakage power under process variations, including spatial correlations," in *Design Automation Conference, 2005. Proceedings. 42nd*, 2005, pp. 523-528.

CHAPTER 5

CELL LIBRARY IMPLEMATATION AND EXPERIMENTAL RESULTS

5.1 Introduction

In this chapter, the implementation of the cell library and the corresponding experimental results will be outlined. The methodology to characterize the higher level digital blocks in the cell library will be introduced first in Section 5.2. Having expanded the library to include a variety of cells whose complexity ranges from 1 gate to more than 3000 gates, large circuit designs can subsequently be efficiently constructed and analyzed with respect to the process variation effects. A Computer-Aid Design (CAD) tool has been developed using MATLAB [1] and SIMULINK [2] to implement the proposed cell library. The details of how to use this tool for the analysis of process variation effects on circuit delay and leakage power performance will be described in Section 5.3. Subsequently, in Section 5.4, the accuracy and speed of the proposed technique will be demonstrated on a 2-stage micropipeline circuit, together with the PDF comparison results for all the blocks used in the demonstration circuit. For validation purposes, all the experimental results are compared with SPICE based Monte Carlo data. Additionally, the pipeline circuit has also been analyzed using traditional flattened SSTA and SPA for comparison purposes, which will emphasize the speed advantage of using the cell library approach. The final section concludes with a summary of the work outlined in this chapter.

5.2 Characterization of Higher Level Blocks

Having characterized all the standard cells in the library, any circuit can be constructed and the corresponding delay performance at each circuit output and the total leakage power dissipation distribution can be estimated. However, as the size of present day circuit designs is typically very large which may comprise hundreds of thousands of gates, it is inconvenient and inefficient to design and evaluate the circuits at such a low level. More complicated functional blocks, such as registers, multiplexers, ALU, decoders etc., need to be included in the cell library, so that the process variation effects on circuit performance in terms of delay and leakage power can be analyzed at a higher level of abstraction, namely architectural level.

As described in the previous chapters, the characterization of gate cells is based on the SPICE simulation runs. The SPICE simulator provides a very accurate prediction of the circuit characteristics, but the simulation has been limited to a small circuit, such as logic gates, by its computational time. It is very time consuming to run SPICE simulations for larger circuits, thus using SPICE based sensitivity to characterise higher level digital blocks in the cell library is not feasible. On the other hand, since all the standard gate cells have already been established, the higher level digital blocks can be modelled using SSTA/SPA analysis results from lower level cells, instead of using SPICE runs. Figure 5-1 shows a schematic view of variability aware cell modelling framework, which illustrates process variation effects propagating from transistor level to architectural level.

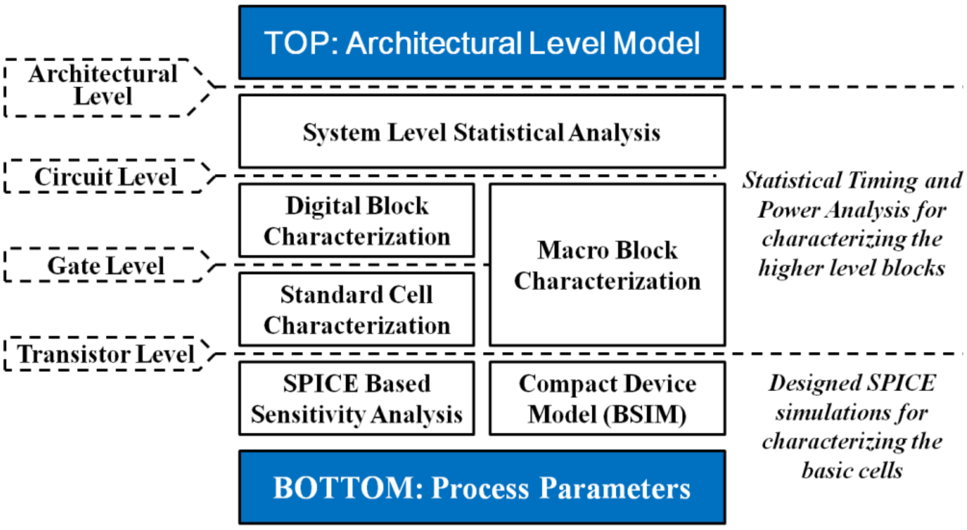

TOP: Architectural Level Model
Architectural Level
System Level Statistical Analysis
Circuit Level
Digital Block Characterization
Gate Level
Standard Cell Characterization
Macro Block Characterization
Statistical Timing and Power Analysis for characterizing the higher level blocks
Transistor Level
SPICE Based Sensitivity Analysis
Compact Device Model (BSIM)
Designed SPICE simulations for characterizing the basic cells
BOTTOM: Process Parameters

Figure 5-1 Schematic view of variability aware cell modelling.

The higher level blocks can be characterised using exactly the same delay and leakage power sensitivity analysis algorithms as the standard cell. The only difference is that the circuit response environment has changed from the SPICE simulator to the variability-aware cells which have already been calibrated in the library. Once a digital block has been characterized, it can be used as the standard cell to perform SSTA/SPA at a higher level in a more complex circuit, expanding the cell library to architectural level blocks in a hierarchical manner. Since only the variability calibrated results of top level digital blocks are used, the models permit a very fast delay analysis to be performed, which also makes it more suitable for scaling up to a larger system. Figure 5-2 shows an example of the library block characterization flow from standard cell to a ripple carry adder.

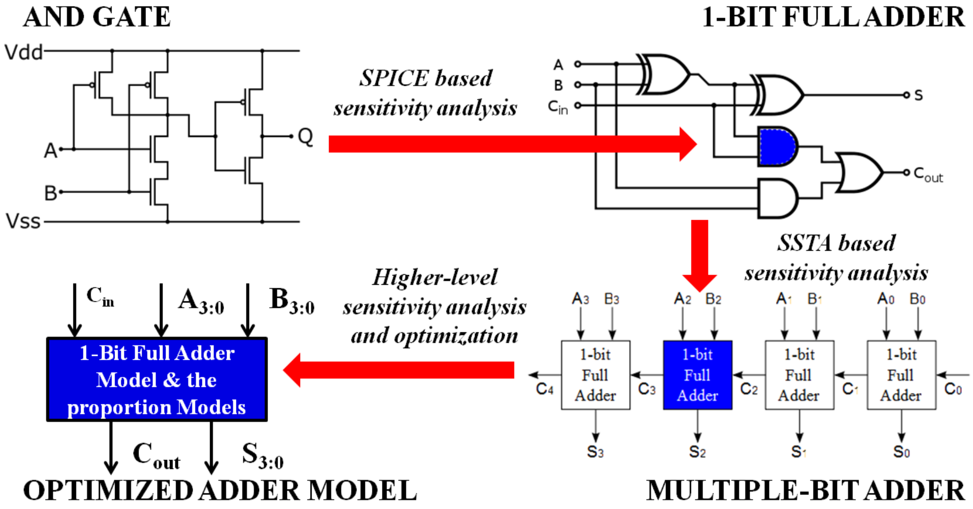


Figure 5-2 Characterization Flow from Standard Cells to 4-bit Adder.

The only problem is that a more complex block has many input terminals, which may lead to a large number of different switching cases. Even though only active switching cases of a circuit block, which are the input signal transitions causing the output signal change in a circuit, need to be considered in the delay modelling, there still could be many of them in some circuits. (If an input signal transition of a circuit causes no change on its outputs, then this transition is inactive and the circuit is assumed to be delay free in this switching case.) Consequently the memory requirement to model the delay distributions in all possible switching cases of a complicated block may be very large. However, larger functional blocks always have a lot of symmetry and multiple occurrences in the circuit. In Figure 5-2, the ripple carry adder is actually a serial connection of multiple full adders, so it can be characterized just by the full adder model. During the work of constructing the whole library, most of the blocks can be represented using a smaller circuit model, the output delay time is simply a matter of the proportions of the model.

On the other hand, the memory space for storing the leakage power models of a larger circuit block is not as much as for storing the delay models. A completed delay model of a block for each switching case is represented by a 3-dimensional LUT as described in Chapter 3, and the leakage power model of a block for each static state is just a vector. Even though all the possible states of a block need to consider for characterizing its leakage power performance, each LUT for leakage model is much smaller than the one for the delay model, thus it will not consume too much memory resources. Furthermore the output signals of each block used in a circuit will be propagated together with the leakage power models. Therefore all gates in the circuit are set into the correct states making the estimation of the circuit leakage power distribution as accurate as possible.

Figure 5-3 and Figure 5-4 show the flow chart for constructing the cell library. Firstly, a library contains all the commonly used standard cells needed to be constructed. It begins with identifying the sensitive device and environmental parameters under the effects of process variations, which will be propagated to high level of design abstraction and interfere with the reliability of circuit performance parameters, such as delay and leakage power dissipation. In the proposed cell library, theses parameters are assumed as Gaussian variables and their variation specification should be predefined before the circuit simulation.

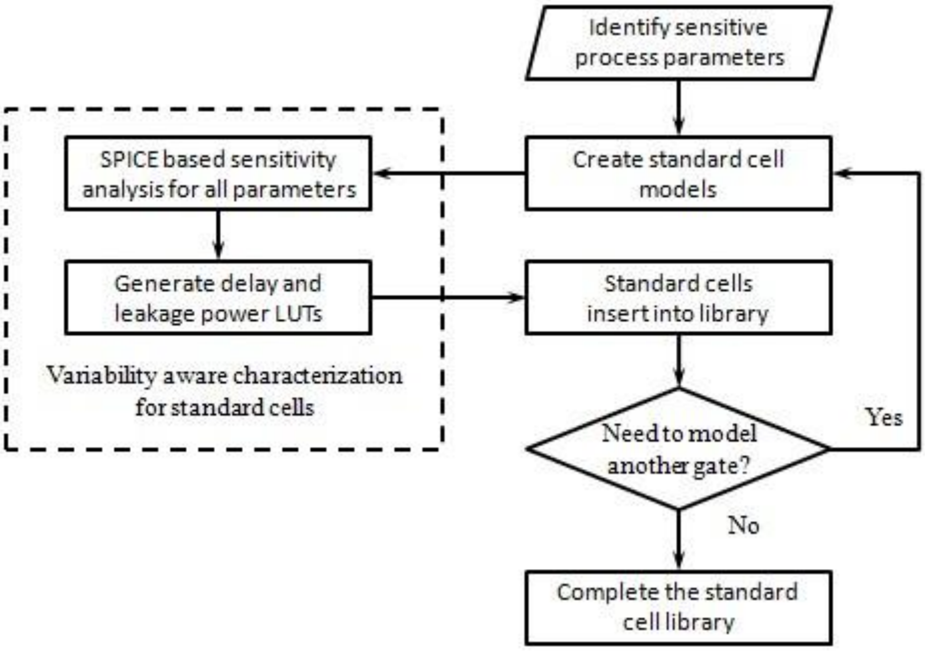

Identify sensitive process parameters
Create standard cell models
SPICE based sensitivity analysis for all parameters
Generate delay and leakage power LUTs
Variability aware characterization for standard cells
Standard cells insert into library
Need to model another gate?
Yes
No
Complete the standard cell library

Figure 5-3 Flow chart for constructing the standard cell library.

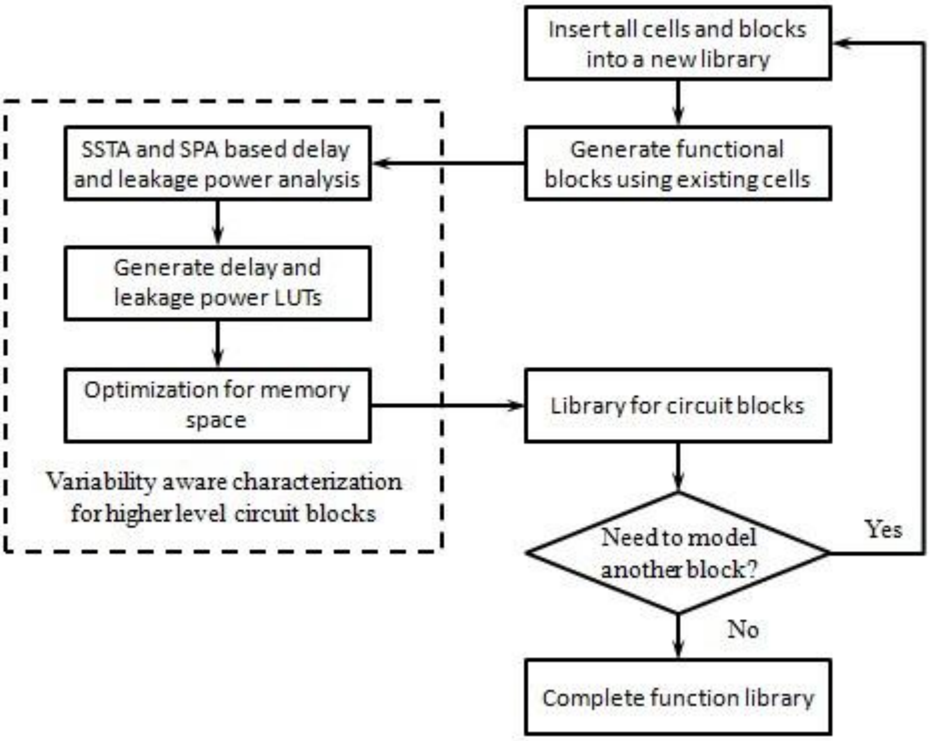

Insert all cells and blocks into a new library
Generate functional blocks using existing cells
SSTA and SPA based delay and leakage power analysis
Generate delay and leakage power LUTs
Optimization for memory space
Variability aware characterization for higher level circuit blocks
Library for circuit blocks
Need to model another block?
Yes
No
Complete function library

Figure 5-4 Flow chart for constructing the function library.

As described in Chapter 2, the process variation effects on the device parameters can be extracted from the process-to-device analysis using DoE and RSM techniques in a TCAD simulation environment. The compact models for NMOS and PMOS transistors can be subsequently generated which contains the mean and sigma values for a variety of device parameters. However, the process-to-device variation analysis is omitted in this work, the variances of the desired device and environmental parameters are assumed known and can be user defined in the cell library.

Having identified the desired parameters whose variation effects on the circuit delay and leakage power dissipation need to be considered, the next step, as shown in Figure 5-3, is to create the standard cells. The sensitivity of each identified parameter with respect to the circuit delay and leakage power dissipation can be derived using SPICE based sensitivity analysis as described in Chapters 3 and 4. The distribution of circuit delay and leakage power dissipation can be modelled as a weighted sum of the device and environmental parameter variables by their sensitivity factors. These sensitivity values will be stored in terms of multiple LUTs for each standard cell. The delay LUTs need to be generated considering different input signal slope and output load capacitance values in each switching case. The leakage power LUTs need to be generated for different static cell states. The SSTA and SPA protocols are also needed in characterizing the standard cells for propagating their delay and leakage power distributions to other cells in the next stage of a circuit. As shown in Figure 5-3, the SPICE based sensitivity analysis of different standard cells will be continually executed until there is no other gates need to be characterized. A standard cell library has now been created which can be used to build up any desired circuit and subsequently analyze the effects of process variation on its delay and leakage power performances.

In order to analyze the process variation at a higher level of design abstraction, a function library comprising more complicated digital blocks needs to be characterized as shown in Figure 5-4. The created standard cell library in Figure 5-3 will become a sub-library and the foundation models for the new function library. New digital blocks will be characterized using the existing cells in the library using the same methodology as for the standard cell, but in an SSTA and SPA simulation environment. The SPICE simulator will lose its computational efficiency when performing sensitivity analysis for a larger circuit. Memory space optimization may be needed for some circuit blocks with a large number of inputs which requires more LUTs to consider all possible delay and leakage power distributions. The number of LUTs required for the circuit blocks with more symmetry and identical subcircuits can be optimized to be very small, thus a large amount of memory space can be saved. Each characterized digital block will be inserted in the function library and used to create new blocks. Circuit blocks with different complexities will be built up step by step. Smaller blocks can be modeled using gate cells, and larger blocks can be modeled using smaller blocks, and so on. Therefore, the function library can thus be expanded until all the necessary blocks for the desired circuit designs are included in it.

Figure 5-5 shows the entity of a circuit block in the proposed function library, which illustrates a general view of its input and output terminals. Like the circuit blocks in other CAD tools, the library cell should be able to perform its normal functionality when needed in a circuit. Therefore, the input and output pins of the Boolean digital signals are included in each block in the library. Furthermore, in order to analyze the process variation effects on circuit delay, the arrival time distributions of all the input signals are also required for a circuit block. The input signal slopes and output load capacitances need to be specified to address the correct delay models inside the block. Based on the SSTA protocol and the corresponding built-in delay LUTs of the block, the delay distribution of each output signal can be computed. On the other hand, the leakage power distribution of the block can be generated based on the Boolean signal inputs and the built-in leakage power LUTs. Each static state of the block corresponds to an independent leakage power LUT, which can be outputted for further SPA. Finally, each output signal of a functional block in the library needs to be outputted together with its transition times for the SSTA in the next stage of a circuit.

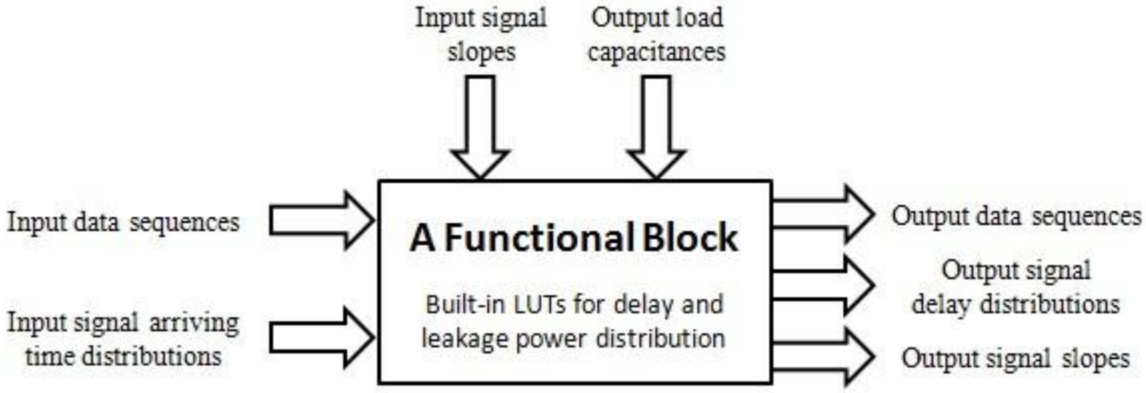

Input signal slopes
Output load capacitances
Input data sequences
Input signal arriving time distributions
A Functional Block
Built-in LUTs for delay and leakage power distribution
Output data sequences
Output signal delay distributions
Output signal slopes

Figure 5-5 Circuit block entity in the library.

All the main cells which have been characterized in the proposed cell library are listed in Table 5-1, with the complexity ranging from a single gate to more than 3272 gates. For demonstration purposes, a 2-stage pipeline circuit will be analyzed using this cell library and the experimental results will be shown and discussed in Section 5.4

Cell Type	**Cell Name**	**No. of gates**	**No. of transistors**
Logic Gate	Inverter	1	2
	2-input and 3-input NAND gate	1	4 / 6
	2-input and 3-input NOR gate	1	4 / 6
	2-input and 3-input AND gate	1	4 / 6
	2-input and 3-input OR gate	1	4 / 6
	2-input XOR gate	1	10
	Buffer (The length is user defined)	variable	variable
Storage Element	D-flip-flop	6	34
	T flip flop	6	36
	Register (The word length is user defined)	variable	variable
	16x16 Register File	2370	9430
Decoder	3-8 Decoder	21	112
	4-16 Decoder	46	243
Multiplexer	4-1 Multiplexer	12	42
	8-1 Multiplexer	28	96
	16-1 Multiplexer	60	198
Asynchronous Element	Muller-C Element	1	6
	Asynchronous Switch	2	10
	Capture and Pass Latch	3	36
	Toggle Element	13	84
	Pipeline Register	46	247
Other Blocks	16-bit Adder	178	832
	ALU	578	2788
	2-stage pipeline circuit	3272	18902

Table 5-1 Main blocks in the cell library.

5.3 Using the Cell Library and the Corresponding Tools

The proposed cell library is implemented in MATLAB and SIMULINK. MATLAB (matrix laboratory) is a numerical computing environment and fourth-generation programming language. It allows matrix manipulations, plotting of functions, processing data and implementation of algorithms, which are very useful and convenient in developing the cell library. SIMULINK is a commercial tool for modeling, simulating and analyzing multidomain dynamic systems. The main advantage of using SIMULINK in developing the cell library is that it provides a graphical block diagramming interface and a set of customizable block libraries, which permits the visualization of all modeled functional blocks and the schematic of the circuit. On the other hand, SIMULINK shares the processing environment of MATLAB and can either drive MATLAB or be scripted from it. Therefore, a set of MATLAB functions can be created for the presetting and post data processing of the circuit analyzed using the cell library.

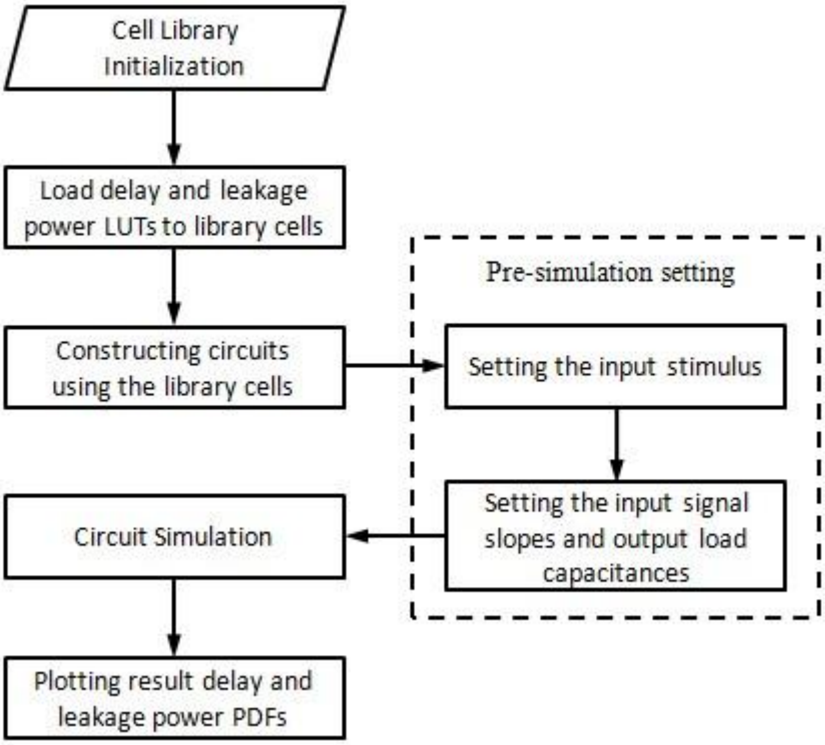

Cell Library Initialization
Load delay and leakage power LUTs to library cells
Constructing circuits using the library cells
Pre-simulation setting
Setting the input stimulus
Setting the input signal slopes and output load capacitances
Circuit Simulation
Plotting result delay and leakage power PDFs

Figure 5-6 Flowchart for the analysis of process variation effects using the cell library.

Figure 5-6 shows the flowchart for the analysis of the process variation effects on circuit delay and leakage power dissipation using the cell library. It begins with an initialization process for setting up the circuit simulation environment. The LUTs for all the library cells will be subsequently loaded. The following step is to build up a circuit using the existing library cells, and set the pre-simulation parameters, such as input signal stimulus, input signal slopes and output load capacitances. The delay distribution of each circuit output signal and the leakage power distribution of the whole circuit can be generated after simulation, and the PDFs can be plotted using the corresponding functions in the proposed cell library tool set. In the following subsections, details about using the cell library and a variety of functional tools for the analysis of process variation effects on circuit delay and leakage power performance will be outlined below.

5.3.1 Initialization function "init"

Before using the cell library, several settings need to be initialized. Basically there are 4 sets of parameters that need to be defined by the user in the initialization phase which is achieved using the "*init*" function in the cell library tool suite. These initialization settings are shown below:

(1) Choose which parameters are to be the global variation sources.
(2) Set the range of variation for the global variables.
(3) Choose which parameters are to be the local variation sources.
(4) Set the range of variation for the local variables.

For demonstration purposes, the variation sources comprise 5 device and environmental parameters, namely, supply voltage V_{dd}, operating temperature T, effective transistor channel length $L_{eff,}$ threshold voltage for N-type device V_{thn} and threshold voltage for P-type device V_{thp}. Firstly, the desired global variation sources need to be chosen for the 5 available parameters. Thereafter the range of variation for each global parameter is defined. Normally, the parameter variation should be within 30% of its mean values, if it is over 30%, the library cells can still work but the analysis accuracy cannot be guaranteed. The next 2 steps of the initialization program are exactly the same as the first 2 steps, but define the specification for local variables. When the initialization is finished, a MATLAB data file "Ini_data.mat" will be loaded into MATLAB workspace and imported by the cell library automatically. This file comprises the delay and leakage power LUTs for the corresponding library cells which may be used for the further process variation effect analysis. The data file can be modified by users in cases of adding new cells to the library or changing the semiconductor technology nodes. On the other hand, the parameters predefined during initialization process will also be loaded into each circuit block in the cell library, thus only the LUTs for the selected parameters will be active, the LUTs for the unselected parameters will be disabled to improve the computational efficiency.

5.3.2 Circuit construction using the library cell

An example of the SIMULINK cell library interface is shown in Figure 5-7. Since SIMULINK shares the work space with MATLAB, the pre-loaded LUTs for the delay and leakage power models during the initialization can be used for all library cells. As described in the previous subsection, all the user defined parameters are also stored in the MATLAB workspace. Therefore, all this data can be treated globally which leads to a significant saving in memory space.

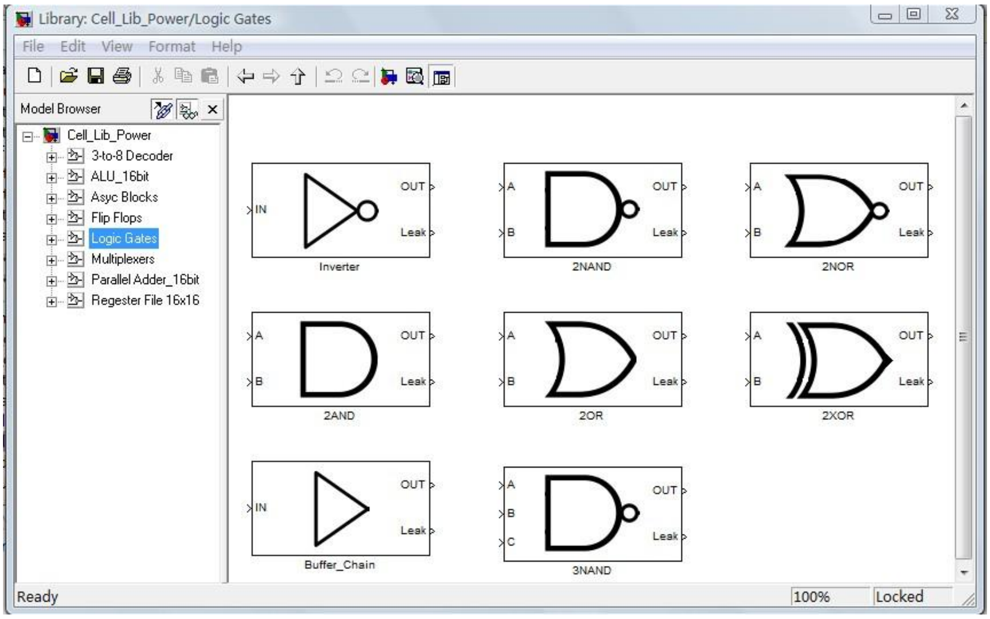


Figure 5-7 Cell library in SIMULINK.

Each cell in the library has an extra output pin, *Leak*, which represents the leakage power dissipation. The library cells can be directly dragged from the library into a new SIMULINK model file to build up any desired circuit. Figure 5-8 shows an example 2-1 multiplexer circuit constructed using the library of standard cells.

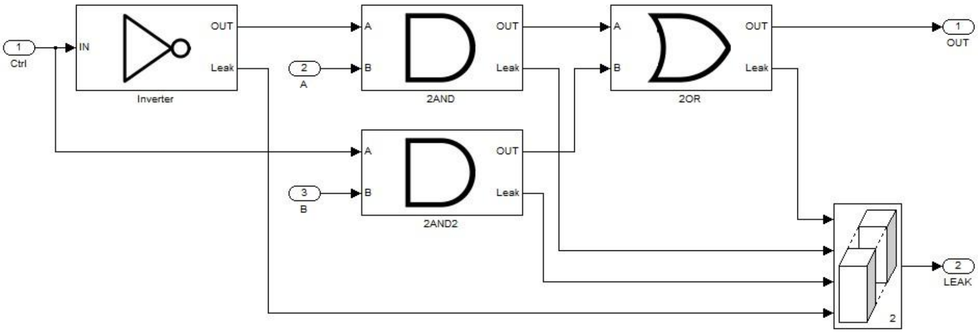


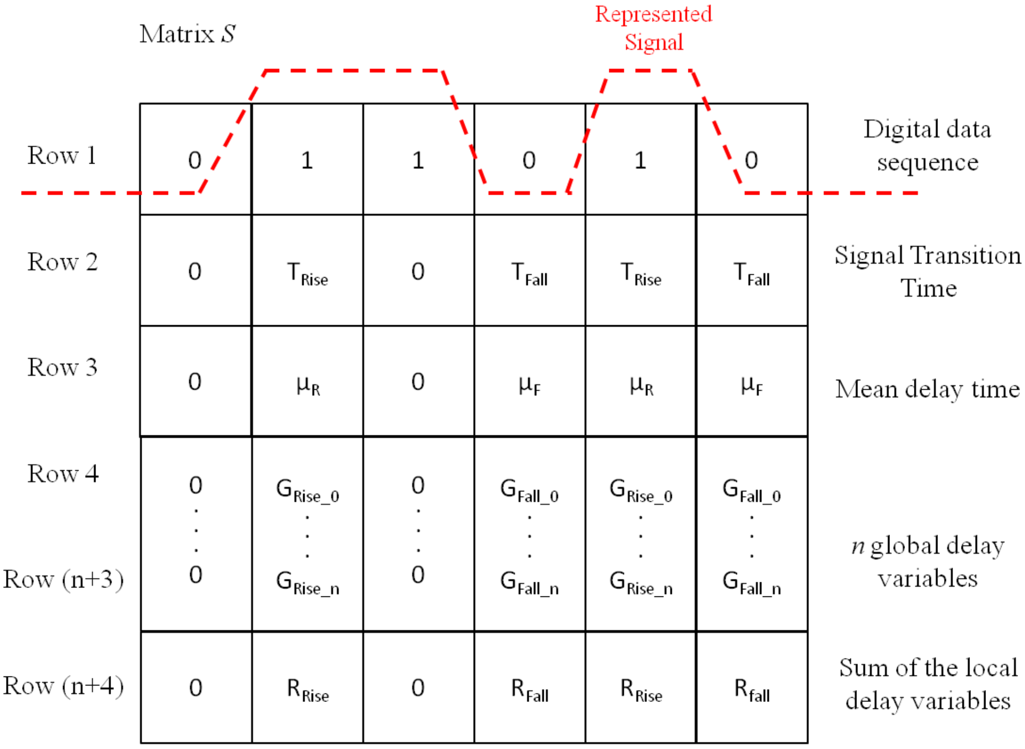
Figure 5-8 Constructing a 2-1 multiplexer using the library cells.

As shown in the above figure, the *leak* terminals of all the cells in a circuit are required to be connected together, then is achieved using the *matrix concatenate* block in SIMULINK. Therefore, the leakage power distributions of all the cells in the circuit will be outputted as a data matrix, where each row represents an independent leakage power model, and the number of the rows is the number of the cells used in the circuit. This matrix will be automatically loaded into the MATLAB workspace, the total leakage power distribution of the circuit will be further computed by summing all the individual power models together in MATLAB. Since the computational speed of MATLAB is much faster than SIMULINK, it is more efficient to take the summation of all the leakage power models in MATLAB workspace rather than in SIMULINK.

5.3.3 Pre-simulation setting

After building up the desired circuit, a number of pre-simulation setting processes are required to specify the input and operating conditions for the simulation. First of all, the input stimulus needs to be defined. All the signals including the input stimulus of a circuit built using the cell library are modeled as a matrix, *S*, as shown in Figure 5-9. The signal matrix contains the information of the Boolean signal data, the signal transition times and the signal arrival time distribution models which are in the same canonical polynomial form as the cell delay models.

The first row of the Matrix *S* represents the digital signal data sequence, in which there are only two legitimate logic values, ‘1’ and ‘0’. The red broken line in the figure is the signal represented by the Row 1 in Matrix *S*. The index of the columns in *S* indicates the corresponding normalized signal timing intervals. Since the circuit timing and power analysis using the cell library only focuses on the delay time during signal transition and the circuit in the static state, the time difference between two adjacent digital signal values is not important. However, a real time signal can be defined in MATLAB and SIMULINK if it is necessary. The second row of the signal matrix shown in Figure 5-9 is the signal transition time for the corresponding digital data in the first row. If there is no transition for the data located at *S(1, x)*, the value *S(2,x)* should be zero.

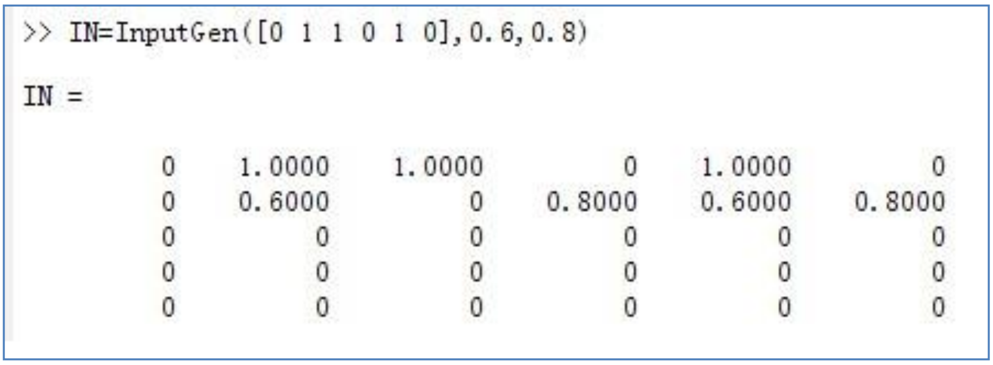


Row 1	0	1	1	0	1	0	Digital data sequence
Row 2	0	T_{Rise}	0	T_{Fall}	T_{Rise}	T_{Fall}	Signal Transition Time
Row 3	0	μ_R	0	μ_F	μ_R	μ_F	Mean delay time
Row 4 Row (n+3)	0 ⋮ 0	G_{Rise_0} ⋮ G_{Rise_n}	0 ⋮ 0	G_{Fall_0} ⋮ G_{Fall_n}	G_{Rise_0} ⋮ G_{Rise_n}	G_{Fall_0} ⋮ G_{Fall_n}	*n* global delay variables
Row (n+4)	0	R_{Rise}	0	R_{Fall}	R_{Rise}	R_{fall}	Sum of the local delay variables

Figure 5-9 Matrix representation of the digital signals for library cells.

The coefficients of the signal arrival time model are stored from the 3rd row to the last row of the Matrix *S*. The 3rd row of Matrix *S* represents the mean delay time of the signal, the last row represents the combined local delay variables of the signal arrival time distribution for the corresponding transition, and the rest of the rows from Row 4 and Row (n+3) are for storing the *n* global delay variables, where *n* is user defined value during the initialization process, in this case the maximum value of *n* is 5 (V_{dd}, L_{eff}, T, V_{thp} and V_{thn}) as described in the previous subsection.

The input stimulus for a circuit should be in the same matrix format shown in Figure 5-9. However, since this matrix is for the primary inputs of a circuit, since no variation exists, all the values below Row 2 of the matrix should be zeros. The function "*InputGen*" in the cell library tool set is used to generate the input matrix of a circuit under simulation, which defines the primary input signal sequences and their slopes. Figure 5-10 shows an example of using "*InputGen*" to generate a "0 1 1 0 1 0" data sequence, as a primary input stimulus to a circuit, with a rising transition time equal to 0.6ns and falling transition time equal to 0.8ns.



```
>> IN=InputGen([0 1 1 0 1 0],0.6,0.8)

IN =

         0    1.0000    1.0000         0    1.0000         0
         0    0.6000         0    0.8000    0.6000    0.8000
         0         0         0         0         0         0
         0         0         0         0         0         0
         0         0         0         0         0         0
```

Figure 5-10 Using "*InputGen*" to generate input signal for a circuit under simulation.

Additionally, in order to make the statistical timing analysis result as accurate as possible, the load capacitance of each cell in a circuit needs to be defined. As the load capacitance of a cell is the sum of the input capacitances of the fan-out cells, the input capacitance values of all input terminals of all cells in the library are also stored in the initialization data file, which has already been loaded into the workspace in MATLAB. This makes it easier when setting the load capacitance of each cell. A meaningful constant name can be used rather than input a capacitance value which probably needs to be evaluated in other CAD tools, such as SPICE. Additionally, the mathematical expressions, such as addition, can be directly used to combine multiple fan-out load capacitances when setting the cell load condition in a circuit.

5.3.4 Circuit simulation and PDF plotting

Having finished all the pre-simulation setting, the circuit built using the library cells is ready for the simulation. To illustrate the analysis principle of the process variation effects on circuit delay and leakage performance, the block diagram of an example of a 2-input NAND gate, shown in Figure 5-11, will be considered. All the circuit blocks in the cell library are constructed in a similar structural manner to the NAND cell, but with different signal routing and LUT addressing protocols corresponding to their functionalities.

As shown in the figure, there are 2 inputs and 2 output terminals for the NAND gate cell. *A* and *B* are the two input signals, and the *Out* is the output signal of the NAND gate. All these 3 signals are in a data matrix form as shown in Figure 5-9. The *leak* terminal, as described before, will output the gate leakage power distribution model. The gate cell is generally divided into two parts, namely, leakage power and delay segments which are used to generate the leakage power and delay distributions due to the selected process parameter variations respectively.

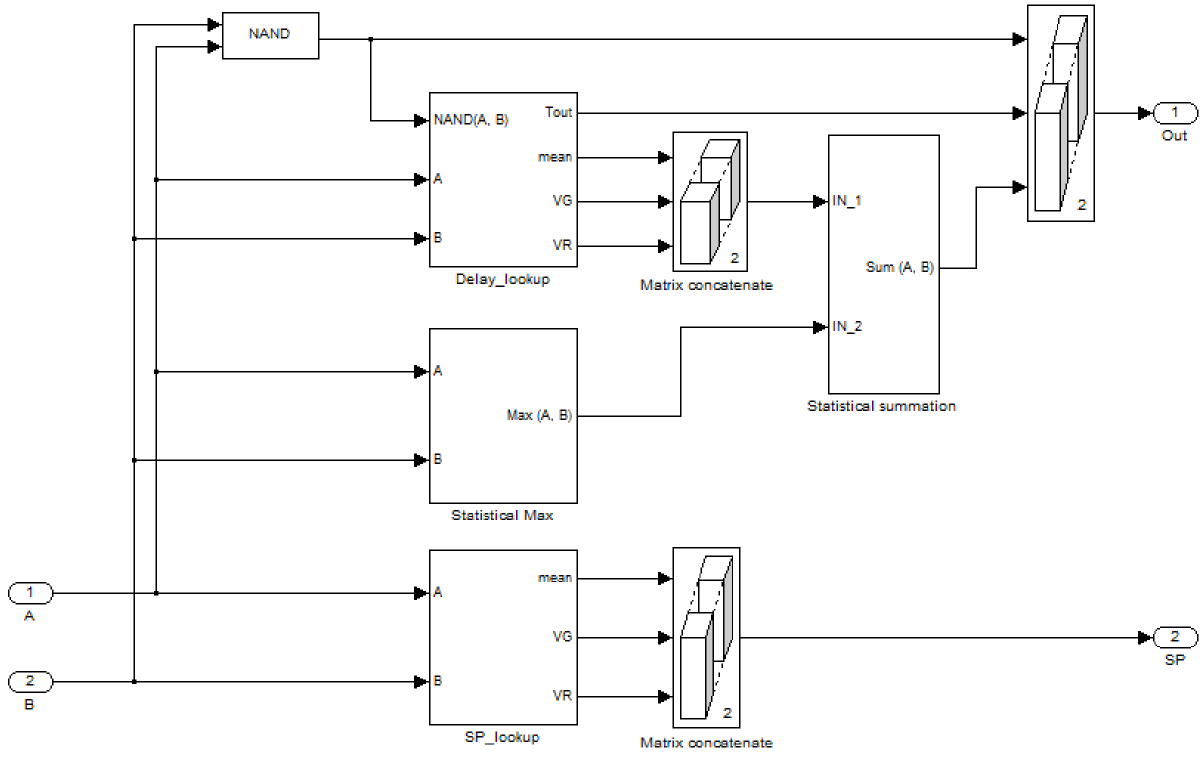


Figure 5-11 SIMULINK model file of a 2-input NAND gate.

As shown in Figure 5-11, the leakage power segement includes a static power LUT block (*SP_lookup*) and a *matrix concatenate*. When signal *A* and *B* arrive at the NAND gate cell, they will be assigned to the *SP_lookup* block. Subsequently, the corresponding canonical leakage power polynomial comprising the values of mean leakage power and the coefficients for the selected global and local variation sources defined during the initialization process, will be addressed from all the possible gate leakage power models according to the input signals *A* and *B*. The *matrix concatenate* block is used to combine the individual components of the leakage power model into a complete model in a matrix form, as the output power distribution of the cell at its *Leak* terminal. The output leakage power distributions of all the cells in a circuit are in the same matrix form, and they will be summed together using SPA technique to calculate the total leakage power distribution of the circuit after simulation.

The delay segement in the NAND gate cell is more complicated than the leakage power segement, which comprises the *NAND* gate function block, delay LUT block (*Delay_Lookup*), statistical max block and statistical summation block. The *NAND* block in Figure 5-11 is used to generate the cell output digital signal, which is the first row of the signal matrix as described in Section 5.3.4. The output values also play an important role in the protocol of delay LUT selection for the whole cell, since output transition case not only depends on the input signals, but also the previous output value. Therefore, both the input and output signals of the *NAND* block are assigned to the delay LUT block. The output delay distribution of the cell should be the statistical sum of the cell delay and the input signal arrival time, since the input of the cell may not be from a primary input and a number of cell delay times have already been accumulated on the input signal. When there are multiple input signals merging into a cell, a statistical maximum operation, achieved using the *statistical max* block, is needed to combine the timing distributions of multiple input signals of the cell into a single canonical delay model. Subsequently, the input timing distribution models and the addressed cell delay models can be summed via the s*tatistical summation* block to compute the final output signal arrival time distribution of the NAND cell. Additionally, another *matrix concatenate* is used to combine the output timing model with the output values and the output signal transition times together into the standard signal matrix at the final stage of the cell construction.

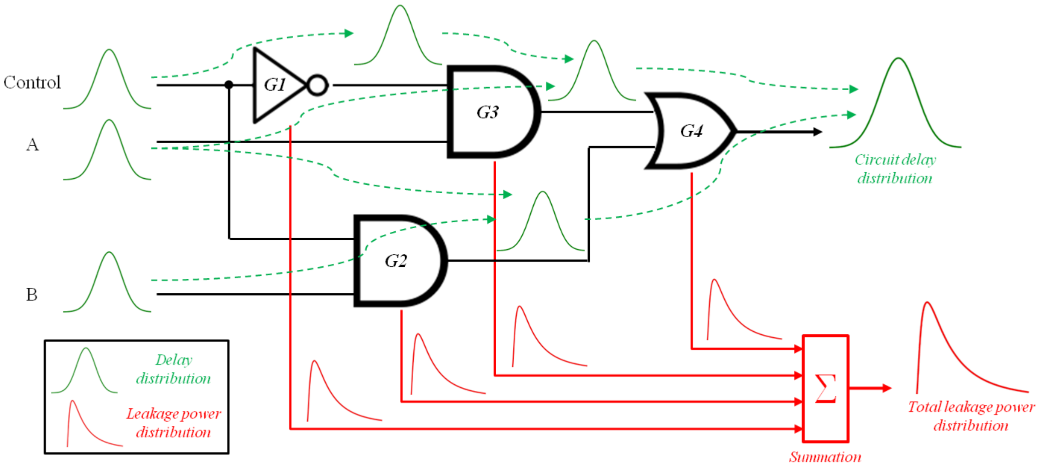

Control
A
B
G1
G3
G4
G2
Circuit delay distribution
Delay distribution
Leakage power distribution
Σ
Summation
Total leakage power distribution

Figure 5-12 A general view of the simulation of a 2-1 multiplexer circuit.

All the circuit blocks in the cell library are built in the same structural manner as Figure 5-11, then the delay and leakage power distributions can be propagated in a similar way through the circuit during simulation so that the process variation effects on them can be analyzed. Figure 5-12 shows a general view of the simulation of a 2-1 multiplexer circuit built using the library cells in SIMULINK.

The red-coloured part of the above figure represents the summation of leakage power distribution using SPA technique for all the cells in the circuit, and the green-colored part indicates the timing distribution propagation through the circuit using block-base SSTA approach during simulation. On the other hand, the functionality of the circuit can also been simulated using the proposed cell library tool set.

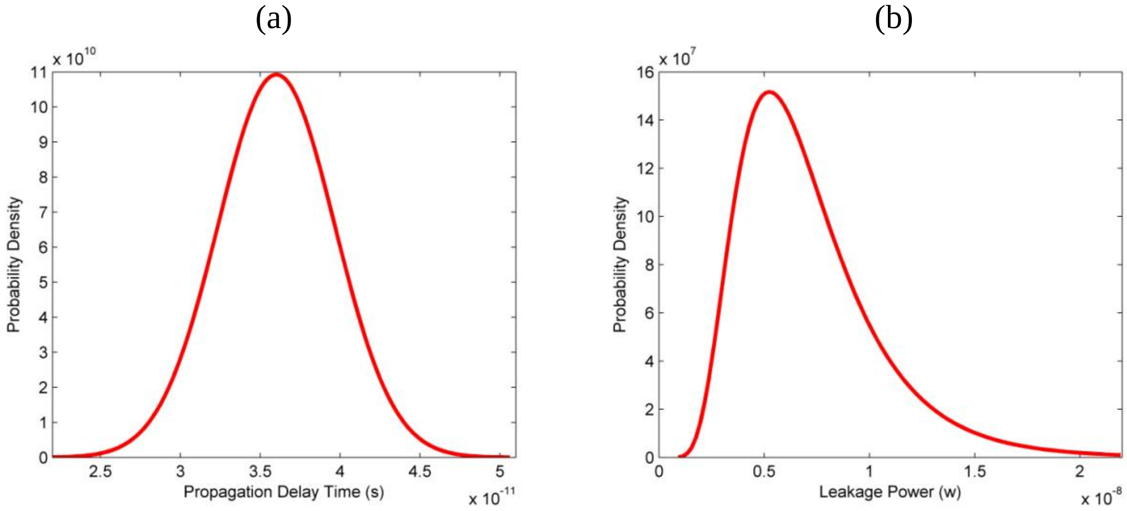


Figure 5-13 Simulation results of the 2-1 multiplexer circuit (a) Delay PDF plot; (b) Leakage power PDF plot.

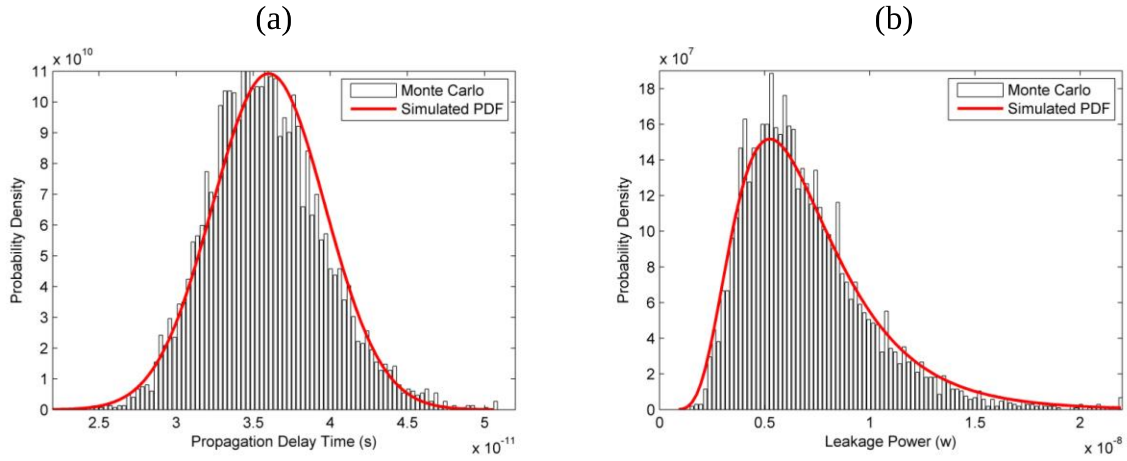


Figure 5-14 PDF plots comparison with MC data (a) Delay distribution; (b) Leakage power distribution.

The delay and leakage power distribution of a simulated circuit cannot be viewed directly in SIMULINK, since they are in a data matrix form. However, these parametric models can be transferred from SIMULINK into the MATLAB workspace. The functions "*PlotDelay*" and "*PlotSP*" in the cell library tool set can be used to plot the delay and leakage power PDF results of the circuit simulation respectively based on the transferred matrices from SIMULINK. Figure 5-13 shows two examples of PDF plots for the 2-1 multiplexer circuit. Additionally, the SPICE based Monte Carlo results of a circuit can be loaded into MATLAB. The functions "HistDelay" and "HistSP" in the cell library tool set can be used to generate the histogram of delay and leakage power distributions based on the imported MC data. These histograms can be subsequently used to match the corresponding simulated PDF results using the cell library for validation purpose. Figure 5-14 shows the PDF comparison graphs of the same multiplexer circuit.

In this section, the full analysis flow of the process variation effects on circuit delay and leakage power performance using the proposed cell library tools has been outlined. It starts with an initialization process, where the variation sources and the corresponding variation specifications can be defined for every cell in the library. Subsequently, the desired circuit can be constructed using the library cells and a set of pre-simulation settings will be followed, including the generation of inputs signal stimulus and load capacitance specification of each cell in the circuit. The properly setup circuit can be simulated in SIMULINK. The functional output signals and the corresponding delay distributions of the circuit will be stored in a data matrix form. The leakage power distributions will be outputted through a leak terminal. The functional simulation results can be viewed directly in SIMULINK, and the PDF results can be plotted in MATLAB by using the corresponding functions in the cell library tool set. Histograms of MC data can also be generated for validation purpose. In order to illustrate the use of the proposed cell library in a larger circuit, a 2-stage pipeline circuit has been simulated. In the next section, a number of experimental results will be shown, and the accuracy and speed analysis compared with MC and traditional flattened statistical approaches will also be discussed.

5.4 Experimental Results

As the demonstration vehicle, a 2-stage pipeline circuit has been constructed and simulated using the cell library. The experimental results for the main blocks used in the pipeline circuit will be shown and discussed in this section. For initialization purposes, the supply voltage V_{dd} and effective channel length L_{eff} were chosen as the global variation sources; n-type threshold voltage V_{thn} and p-type, V_{thp}, are selected as local variation sources. The mean and 3 sigma values of these parameters are shown Table 5-2. All these variables are assumed to have a Gaussian distribution. All the simulations are run using an Intel dual core 2.0 GHz processor.

Parameter	Mean value	3 sigma value	Variation
V_{dd}	*1 v*	*0.15 v*	*±15%*
L_{eff}	*45 nm*	*9 nm*	*±20%*
V_{thn}	*0.397 v*	*0.0595 v*	*±15%*
V_{thp}	*0.339 v*	*0.0508 v*	*±15%*

Table 5-2 Experimental parameters and the range of variations.

The block diagram of the 2-stage pipeline circuit is shown in Figure 5-15, which is an event-controlled two-phase bounded data system [3]. Both the falling and rising signal transitions can trigger this pipeline circuit. When the request (*req*) signal event from the previous stage is sent to the pipeline cell, an acknowledgement (*ack*) signal event will be generated back to the previous stage. Simultaneously, a delayed *ack* signal event will be sent to the next stage as the request signal event of the current pipeline cell, and then a new *ack* signal event will be fed back to the pipeline cell from the next stage. Only if both the *req* event from previous stage and the *ack* event from the next stage have been received by the pipeline cell, the new data which is assumed to be ready during the last pipeline cycle will be latched into pipeline registers beginning a new pipeline cycle. The delay element between each pipeline stage must be larger than the delay of combinational logic circuit at each stage in order to make sure all the input data is stable at the beginning of each cycle.

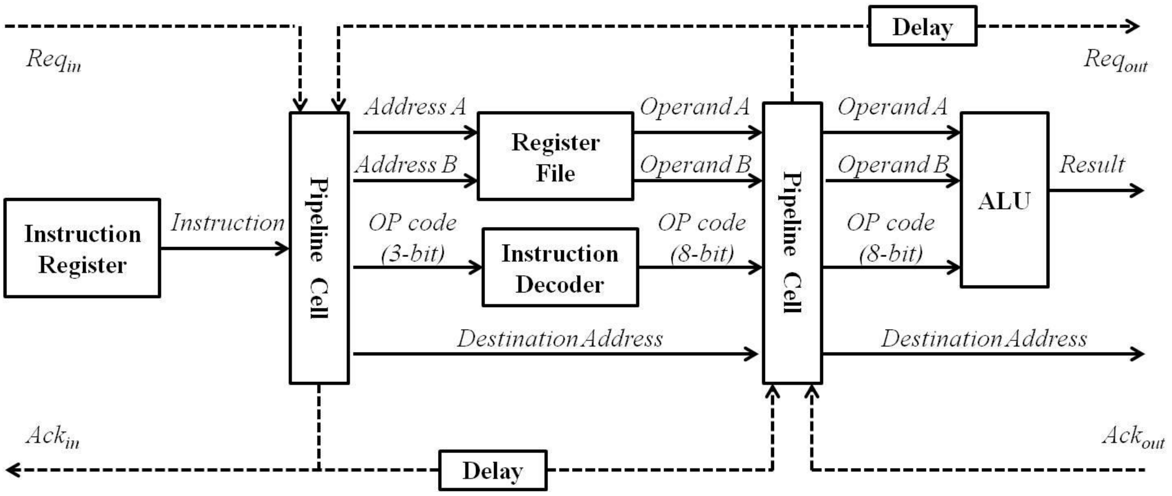


Figure 5-15 Block diagram of the 2-stage pipeline circuit.

In Figure 5-15 the pipeline cell is implemented using Muller-C elements, Toggle elements and Capture-Pass Flip Flops (CP Flip Flop) [1]. The instruction contains two 4-bit addresses for the two operands *A* and *B*, a 4-bit destination address for the ALU result, and a 3-bit operation code. The instruction decoder is used to decode the 3-bit operation code into an 8-bit one-hot *op code* for the ALU; the register file contains 16 registers with 16-bit word width (16x16 bits) and two 4-to-16 address decoders for storing and addressing the operands *A* and *B*; the ALU circuit can execute 8 operations which are addition, left/right shift, rotation, inversion, logic *AND*, logic *OR* and exclusive *OR (XOR)*.

The rest of this section will discuss the experimental PDF results for the delay and leakage power performance of all the blocks used in the pipeline circuit. For validation purposes, the demonstration circuit is also constructed using the SPICE simulator, and the corresponding process variation effects on its delay and leakage power performance are analyzed using the MC simulation with 5000 samples, whose results are used as the reference distributions of the circuit performance parameters. Therefore, the predicted PDFs can be matched with MC data so that the accuracy of the simulation results using the cell library can be verified. The PDF comparison graph will be shown block by block in the following subsections. The numerical results in terms of comparisons of the mean and sigma values between predicted distributions and MC data will be listed at the end of this section, as well as the corresponding computational time for both analysis techniques. Additionally, the computational time for the traditional flattened statistical analysis of the demonstration circuit will also be listed. By comparing these experimental results, it can be shown that the proposed cell library can maintain an acceptable error rate of mean and sigma values, within 5% compared with MC data, for all the circuit blocks used in the pipeline circuit, and a rapid analysis speed which is at least 50 times faster than traditional statistical approaches.

5.4.1 Register

The instruction register in the pipeline circuit is used to store the instruction data for the next processing unit, which is made up of D-flip-flops. The D-flip-flop captures the value of the D-input at a sensitive edge of the clock cycle, such as rising and falling edges, and the captured value becomes the *Q* output. At other times, the output *Q* does not change. Figure 5-16 shows the circuit and symbol of a positive-edge-triggered D-flip-flop. Typically, most D-flip-flops in IC design have the preset and clear function, which forces the output *Q* to be set or reset.

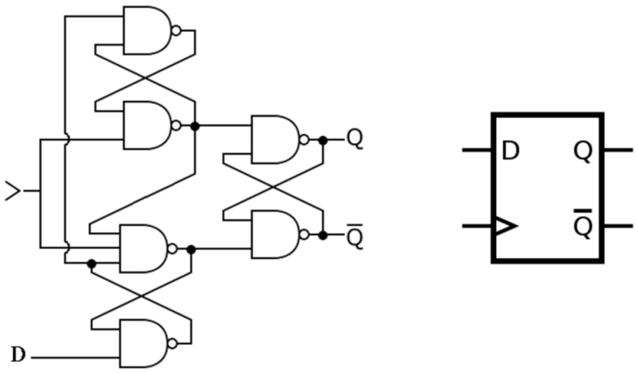


Figure 5-16 A positive-edge-triggered D-flip-flop [4].

For the sequential elements, such as the D-flip-flop, their circuits normally involve several feedback signals. This make the statistical timing analysis more complicated. For a gate in a sequential circuit, the statistical maximum of it inputs signal arrival time distributions need to consider the output signal delay distributions of the same gate, which will be fedback to the input gate terminals in a very short time. Therefore, when the sensitive clock edge arrives, it may require numerous statistical operations to calculate the output delay response using SSTA. On the other hand, the complicated correlations among the internal signals in the sequential circuits make the accuracy of SSTA decrease under the linear assumption of the statistical maximum operations. Additionally, the signal routing protocol for such a complicated block in timing estimation also becomes more difficult to write. In order to improve the timing analysis accuracy and computational efficiency, all the sequential elements in the cell library are modelled as the standard cells, which use the SPICE based sensitivity analysis results to model their delay distributions rather than SSTA using the existing gate cells. Since the typical circuit size of sequential elements, such as flip flops, is quite small (normally comprises 4 to 8 logic gates), running the SPICE simulations on them will not consume too much time.

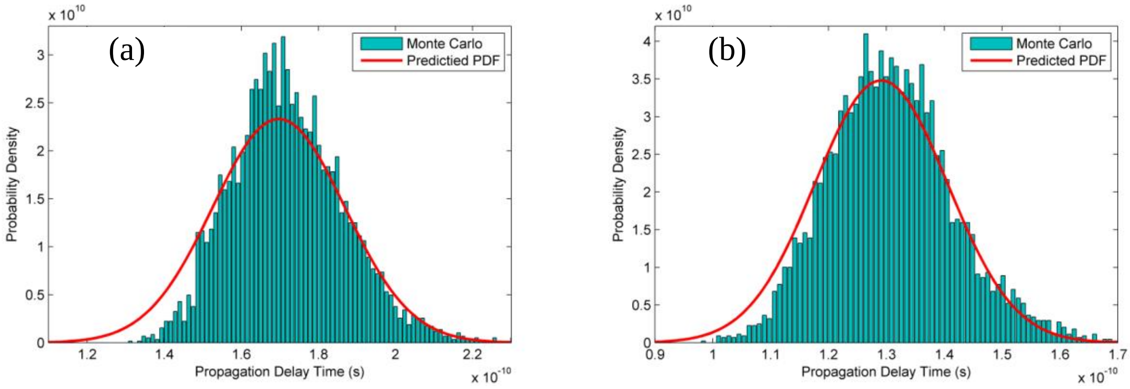


Figure 5-17 Delay PDFs of a D-flip-flop when (a) Q is rising and C_{load}=4 C_{unit}; (b) Q is falling and C_{load}=2 C_{unit}.

Figure 5-17 shows two PDFs comparison graphs between the 5000 sample Monte Carlo simulation results and the predicted delay distributions of output signal Q of a D-flip-flop at different input and load conditions. The output signal Q of the D-flip-flop is at rising and falling transition in Figure 5-17 (a) and (b), and the load capacitance (C_{load}) of Q is set to 4 and 2 unit load values (C_{unit}) respectively, where the unit load is the input capacitance value of a static CMOS inverter circuit which has been introduced in Chapter 3. It can be seen from the above graphs that the predicted delay PDFs and the MC data are well matched, which indicates that the D-flip-flop block in the cell library can accurately model the delay characteristics of the actual circuit. On the other hand, the leakage power model of the D-flip-flop block in the library can also be constructed using SPICE based sensitivity analysis. Figure 5-18 show two leakage power PDF comparison graphs of the D-flip-flop block at different static states.

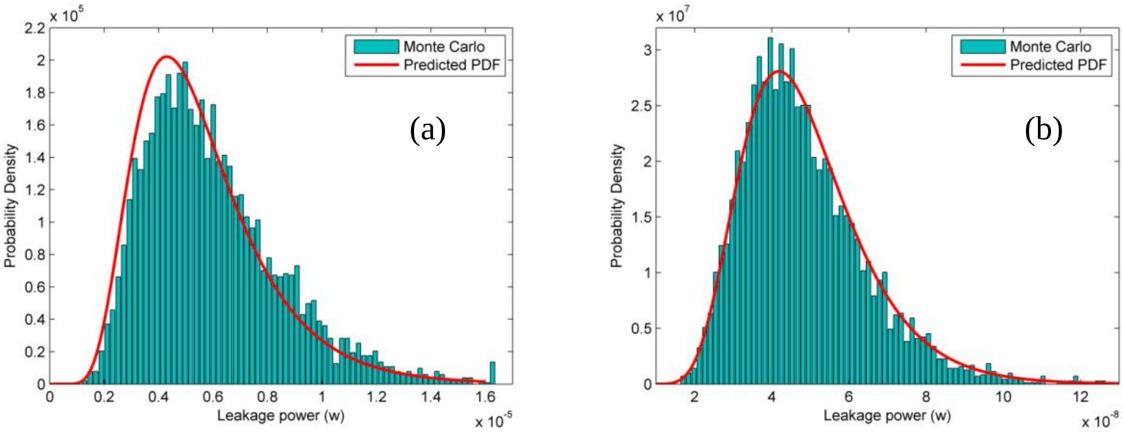


Figure 5-18 Leakage power PDFs of a D-flip-flop when (a) all inputs are at logic high; (b) all inputs are at logic low.

The instruction register in the pipeline circuit is made up of 16 D-flip-flops. The global signals, *clock*, *preset* and *clear*, are shared by all the flip flops in the register. The register is used to store the 16-bit instruction for every pipeline cycle, comprising a 3-bit operation code which defines the task for the following computing process, two 4-bit addresses for the operands, a carry bit and a 4-bit destination address for the computation result.

5.4.2 *Decoder*

The instruction decoder in the pipeline circuit is a 3-8 decoder, which is used to convert the 3-bit operation code into an 8-bit one hot code for the ALU circuit in the next pipeline stage.

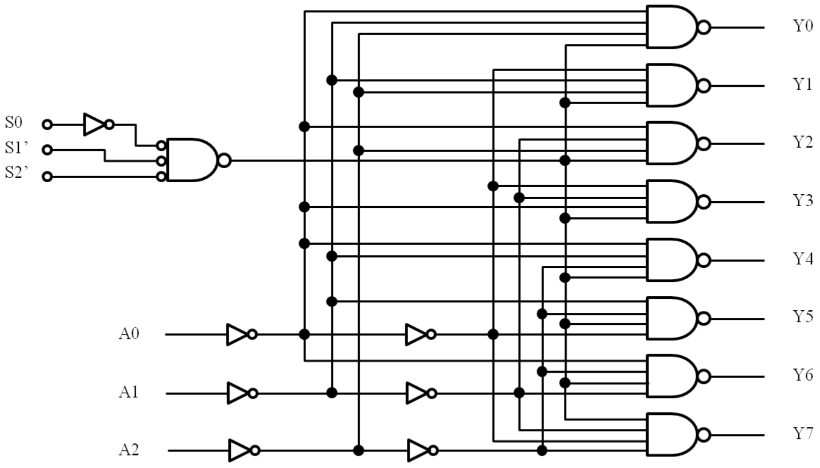


Figure 5-19 The circuit of 3-8 decoder [5].

The decoder circuit is shown in Figure 5-19. The input signals *S0*, *S1'* and *S2'* are used for enable purposes, the decoder is only active when *S0='1'* and *S1'='0'* or *S2'='0'*. The input signals *A0* to *A2* indicate a 3-bit binary number, in which *A0* represents the least significant bit and *A2* represents the most significant bit. The 8-bit output signal from *Y0* to *Y7* indicates the corresponding 8 numbers represented by the binary input.

Figure 5-20 shows the delay and leakage power PDF comparison graphs of the decoder block. The delay distributions are generated at the most significant bit of the decoder output signals (*Y7*). The leakage power distributions are generated when all the input signals of the decoder are at logic high state.

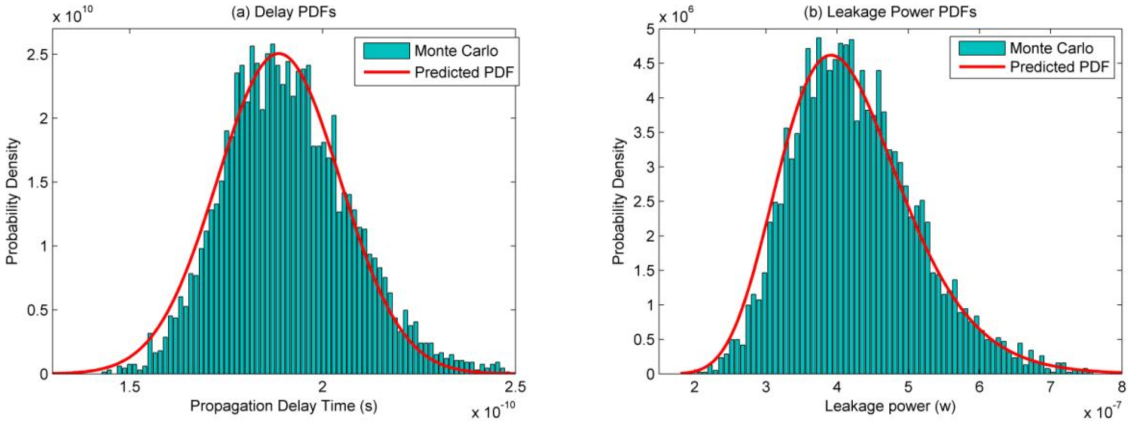


Figure 5-20 (a) Delay PDF comparison graph of a 3-8 decoder; (b) Leakage power PDF comparison graph of a 3-8 decoder.

The 3-8 decoder block in the cell library can be further used to model the 4-16 decoder as the circuit shown in Figure 5-21 which will be used to decode the 4-bit data address for the register file.

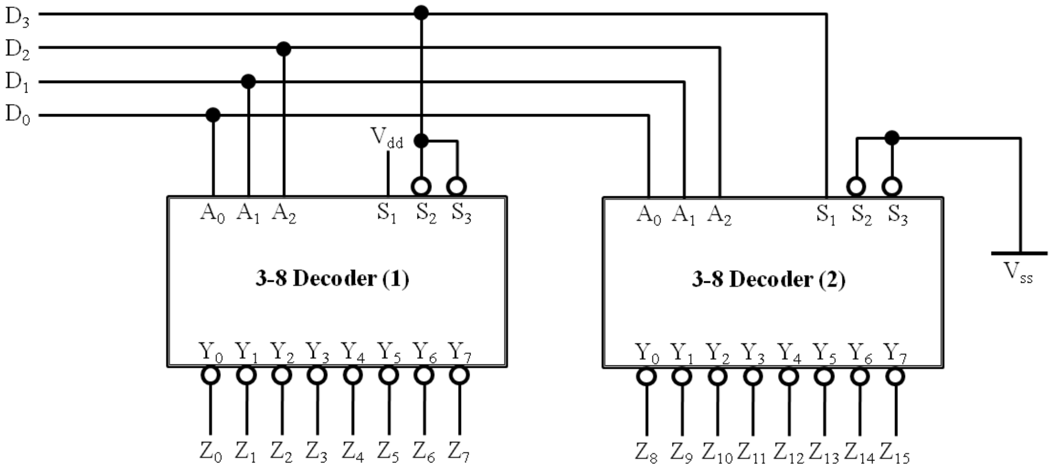


Figure 5-21 Constructing a 4-16 decoder using two 3-8 decoders [5].

5.4.3 Register File

The Register File (RF) block in the pipeline circuit is the made up of 16 registers. Each register can store a 16-bit data. Therefore, it requires 256 D-flip-flops in this block and a 4-bit signal to address each register. Figure 5-22 shows the block diagram of the RF circuit with two 16-bit outputs for exporting stored data to the external circuit (Read) and a 16-bit input for writing external data to the RF (Write). The write enable signal (*WR*) controls the read/write state of the RF: *WR*=0 indicates "Read" and *WR*=1 indicates "Write".

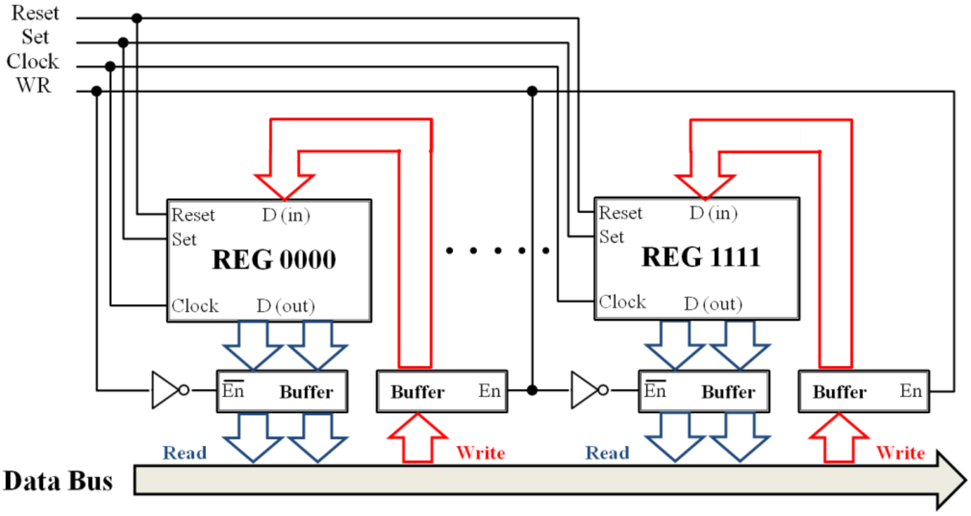


Figure 5-22 Block diagram for the Register File circuit.

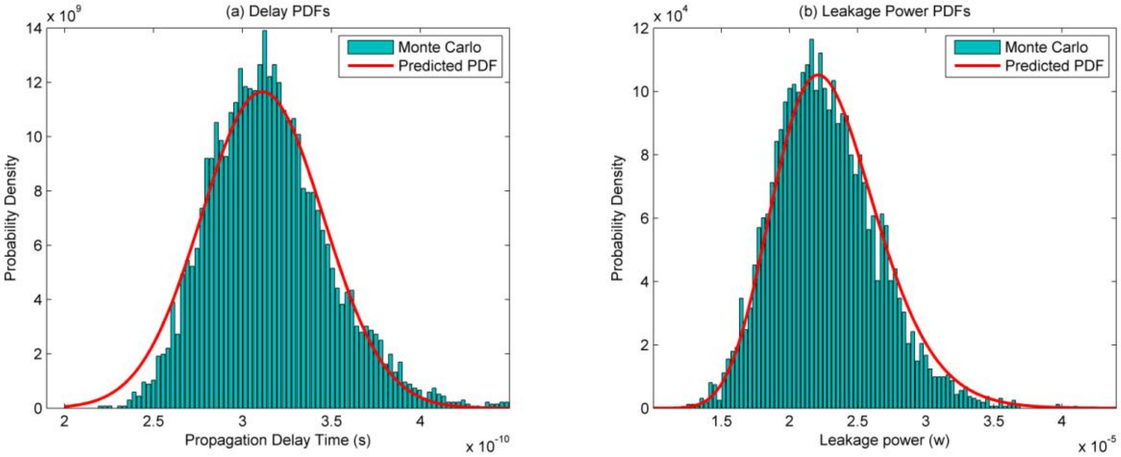
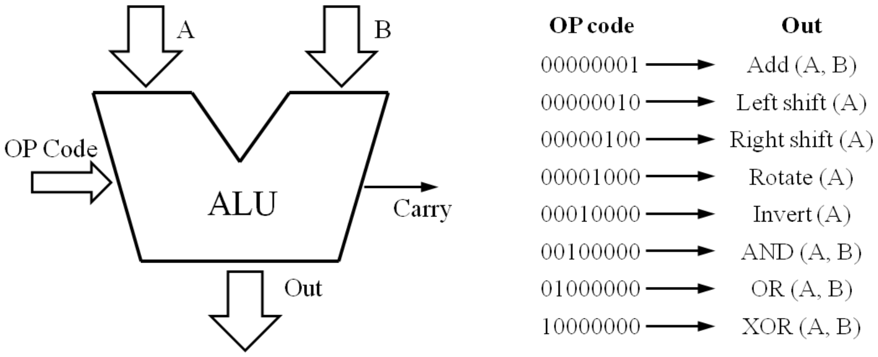


Figure 5-23 (a) Delay PDF comparison graph of the register file; (b) Leakage power PDF comparison graph of the register file.

Figure 5-23 shows the delay and leakage power PDF comparison graphs of the RF block. The delay distributions are generated at one stage of the register when its data is being read. The leakage power distributions are generated when the write enable (*WR*) signal is at logic low state and all the other input signals of the RF are at logic high state.

5.4.4 ALU

The ALU in the pipeline circuit can perform 8 operations as shown in Figure 5-24. It is controlled by an 8-bit one hot operation code. The word length of the two operands and the output data of the ALU circuit is 16 bits. There is also a flag bit at the ALU output as the carry signal for the *Add* operation.



OP code	Out
00000001 →	Add (A, B)
00000010 →	Left shift (A)
00000100 →	Right shift (A)
00001000 →	Rotate (A)
00010000 →	Invert (A)
00100000 →	AND (A, B)
01000000 →	OR (A, B)
10000000 →	XOR (A, B)

Figure 5-24 Block diagram for the ALU circuit.

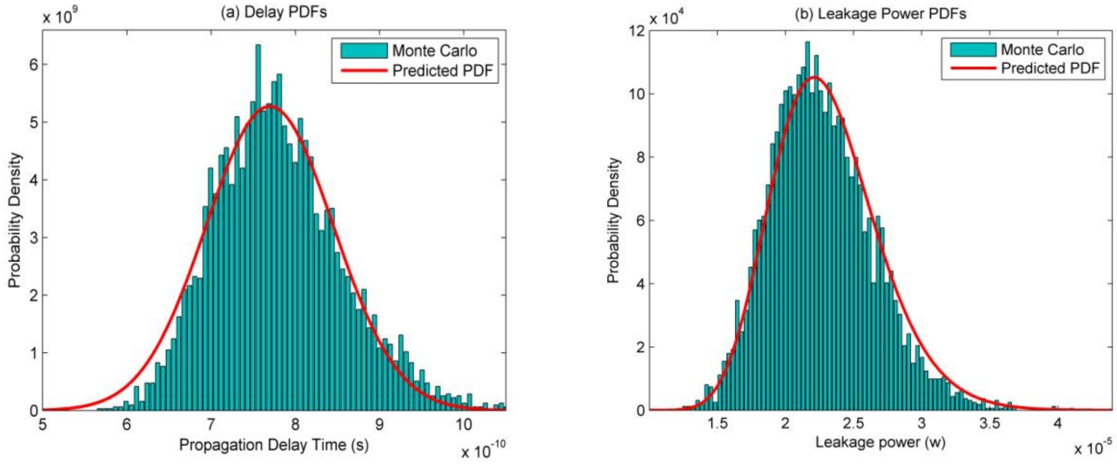


Figure 5-25 (a) Delay PDF comparison graph of the ALU circuit; (b) Leakage power PDF comparison graph of the ALU circuit.

Figure 5-25 shows the PDF comparison graphs for the ALU circuit. The delay distributions are generated at the most significant bit of the ALU output signal when the *add* operation is processed. The active signal path of the adder inside the ALU circuit is longer than the other functional blocks. The leakage power distributions are generated when all the input signals to the ALU circuit are at a logic high state.

5.4.5 *Asynchronous pipeline register*

The demonstration pipeline circuit is an asynchronous system, and the pipeline registers are controlled by signal events rather than the specific sensitive signal edges, such as rising and falling edges. The basic asynchronous cell used in the pipeline circuit is the well-known Muller C-element, whose circuit is shown in Figure 5-26. The Muller C-element acts as an AND element for events. When both inputs to a C-element are in the same logic state, the logic state will be propagated to the output signal. If the two inputs of a C-element differ, it will use the internal storage to retain its previous state and the output value also remains unchanged. The C-element is typically used in the pipeline circuit to capture both events of the request and acknowledge signal so allowing the register to pass data and be processed during a pipeline cycle.

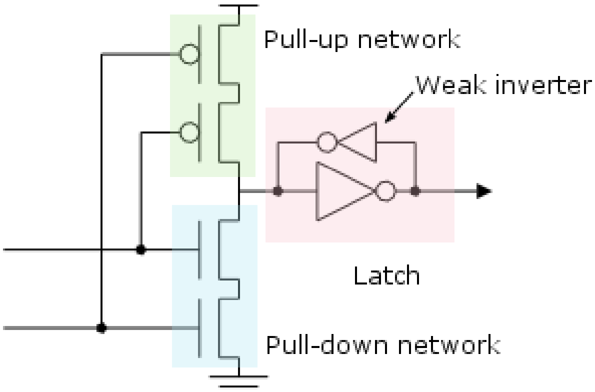


Figure 5-26 Circuit for a Muller C-element [6].

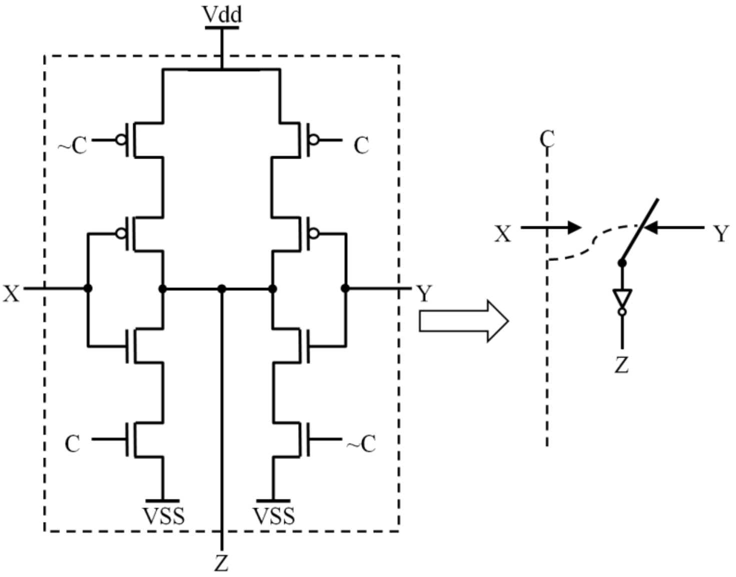


Figure 5-27 Switch element for the asynchronous pipeline register [3].

Figure 5-27 shows another basic element in the asynchronous pipeline circuit, the event-controlled switch [3], which will be further used in constructing an event-controlled storage element. In the transistor implementation of the switch is made of both the true and the complement forms of its control signal, *C* and *~C*, which implies an inversion of the control signal not shown explicitly in the figure. The rising and falling transitions of the control signal *C* will let the input data *X* and *Y* propagate to the output *Z* alternately.

Using the switch circuit, an event-controlled storage element with the pre-clear terminal can be constructed as shown in Figure 5-28, which is also known as CP (Capture and Pass) flip flop [3]. Each pipeline register contains a number of event-controlled storage elements, each flip flop has a clear terminal which can be used to pre-reset the register for initialization purposes. The signals *C (Capture)* and *P (pass)* act like the *request* (*req*) and the *acknowledgement* (*ack*) for the pipeline register in a hand shaking protocol, which are connected to a *XOR* gate. When an event on *C* arrives which indicates a request to transfer data, it will trigger the switch circuits to capture the input data into the flip flops. After a certain timing period, the event on the signal *P* will arrive which indicates the acknowledgement of the data transfer. Subsequently, it triggers the switch circuit back to an internal inverter loop which will let the data pass and be processed during the corresponding pipeline cycle. The toggle element in Figure 5-28, which alternately steers events to its outputs starting with the dot, is used to generate the *capture done (Cd)* and *pass done (Pd)* signals for the handshake signals to the previous and next pipeline stages.

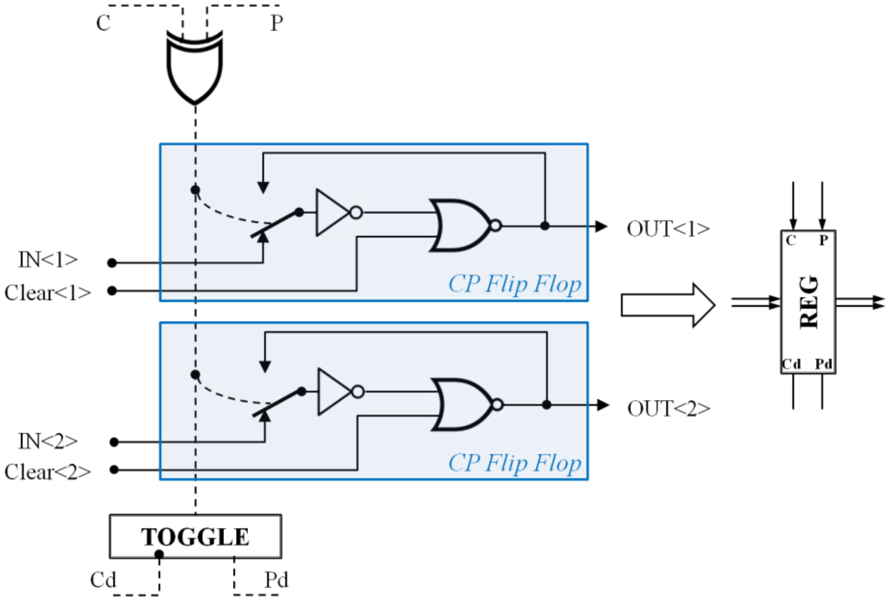


Figure 5-28 Event-controlledled storage register.

Figure 5-29 shows the micropipeline without processing, which is built up using the event-controlled registers. The circuit uses the bundled data interface, in which the delay time between any adjacent pipeline stages should be longer than the required processing time of the combinational logic circuit between two pipeline registers. This allows the registers enough time to capture the valid input data during each pipeline cycle. Figure 5-30 shows the functional simulation result of an event-controlled pipeline register.

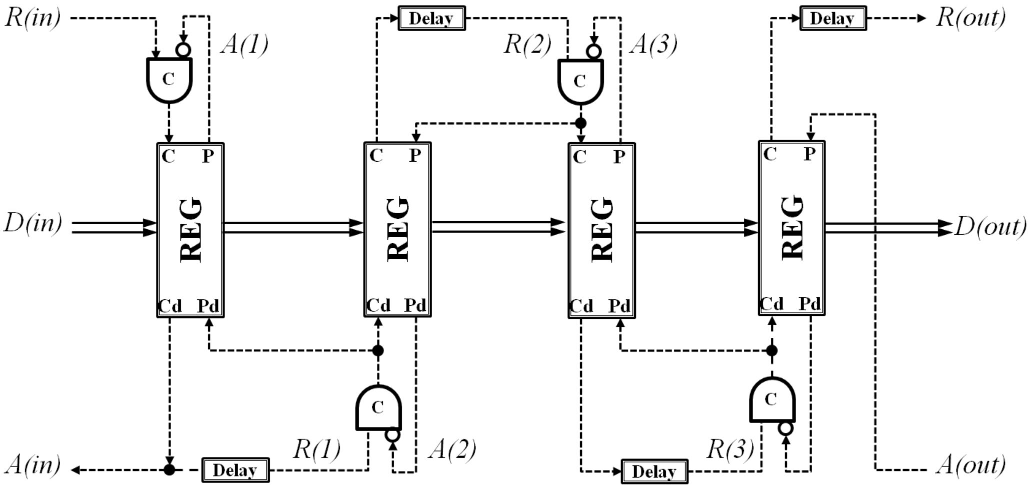


Figure 5-29 Micropipeline circuit without processing [3].

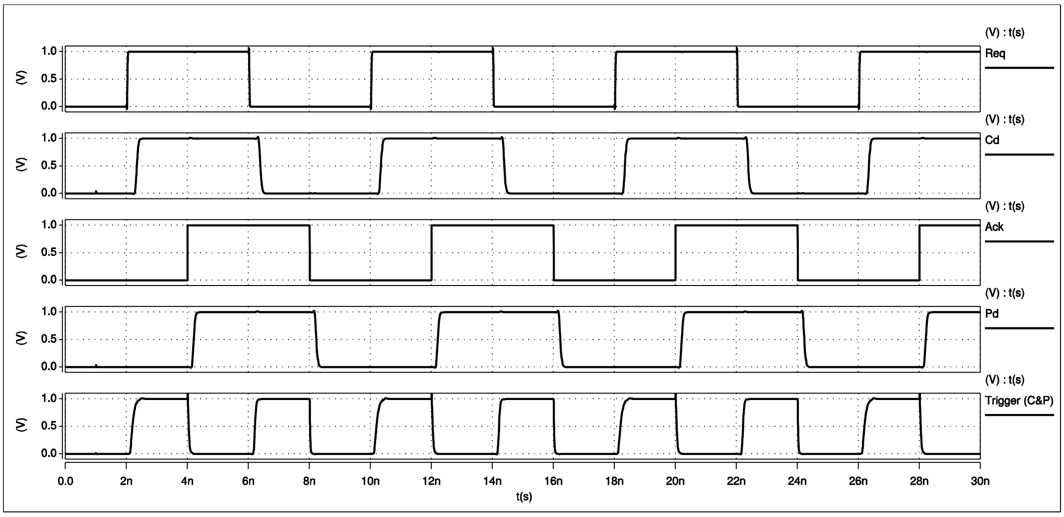


Figure 5-30 Functional simulation result of the asynchronous pipeline register.

Figure 5-31 shows the delay and leakage power PDF comparison graphs for a pipeline register block with a delay element using inverters. The delay distribution analysis results due to the process variation effects of the pipeline register and the corresponding processing circuits can be estimated using the proposed cell library blocks. This will help the designer to decide how much delay is needed in each pipeline stage to make the whole system work without timing errors.

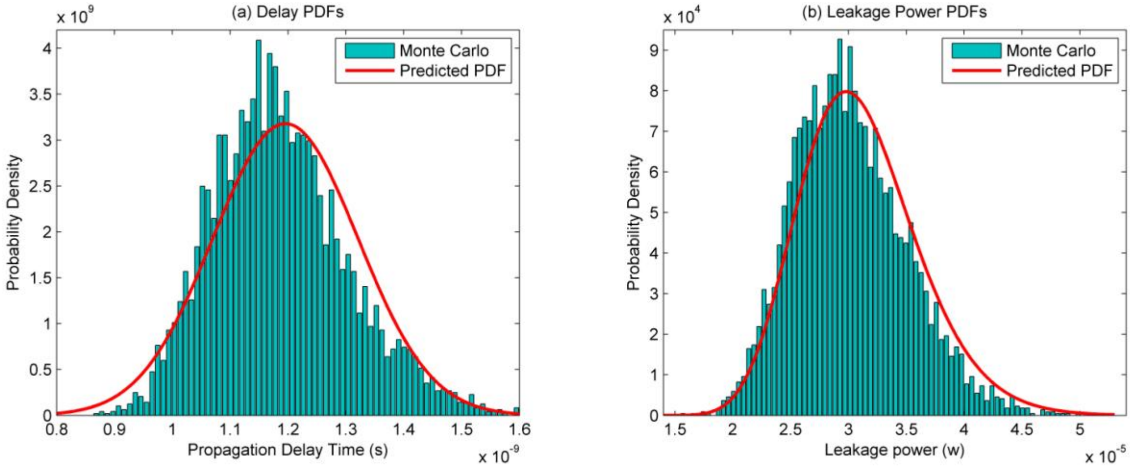


Figure 5-31 (a) Delay PDF comparison graph of the pipeline register; (b) Leakage power PDF comparison graph of the pipeline register.

5.4.6 Analytical experimental results

Table 5-3 and Table 5-4 list the mean/sigma values of the predicted delay and leakage power PDFs using the cell library and the reference PDFs generated by Monte Carlo simulation for the main blocks used in the pipeline circuit, as well as the corresponding error analysis.

Blocks	Number of Gates	Cell Library		Monte Carlo (5000 sample)		Error (%)	
		μ (ps)	σ (ps)	μ (ps)	σ (ps)	μ	σ
Muller C Element	1	146.81	21.60	146.13	21.28	0.46	1.48
D-flip-flop	6	131.76	8.87	133.14	9.30	1.03	4.67
Toggle Element	13	245.53	19.36	249.77	20.27	1.70	4.50
3-to-8 Decoder	21	191.04	12.95	188.64	12.34	1.26	4.65
Pipeline Registers	46	196.42	17.62	197.41	16.35	0.50	4.78
Delay Element	50	988.16	77.65	980.61	82.50	0.77	4.88
Multiplexer	60	525.43	46.67	513.99	45.97	2.22	1.51
Full Adder	178	404.91	22.86	404.78	22.28	0.03	2.25
ALU	578	781.43	61.03	776.61	62.48	0.62	2.32
Register File	2370	318.16	28.32	315.68	27.79	0.79	1.90
Pipeline Circuit	3272	1380.2	98.45	1364.75	96.64	1.14	1.88
Average Error						0.96	3.53

Table 5-3 Delay accuracy comparison with SPICE-based Monte Carlo simulation.

Blocks	Number of Gates	Cell Library		Monte Carlo (5000 sample)		Error (%)	
		μ (w)	σ (w)	μ (w)	σ (w)	μ	σ
Muller C Element	1	10.04 n	3.88 n	10.06 n	4.02 n	0.26	3.48
D-flip-flop	6	48.85 n	16.19 n	48.47 n	16.64 n	0.78	2.70
Toggle Element	13	120.07 n	34.59 n	119.85 n	35.88 n	0.18	3.60
3-to-8 Decoder	21	304.37 n	75.65 n	301.94 n	76.95 n	0.80	1.69
Pipeline Registers	46	522.83 n	136.61 n	525.09 n	140.77 n	0.43	2.96
Delay Element	50	318.49 n	80.23 n	317.95 n	82.11 n	0.17	2.29
16-to-1 Multiplexer	60	706.20 n	171.68 n	702.47 n	176.61 n	0.53	2.79
Full Adder	178	1.24 μ	0.30 μ	1.23 μ	0.31 μ	0.75	1.31
ALU	578	5.60 μ	1.30 μ	5.56 μ	1.32 μ	0.72	1.52
Register File	2370	19.03 μ	4.68 μ	18.77 μ	4.74 μ	1.39	1.27
Pipeline Circuit	3272	27.14 μ	4.86 μ	26.32 μ	4.93 μ	3.12	1.42
Average Error						0.83	2.28

Table 5-4 Leakage power accuracy comparison with SPICE-based Monte Carlo simulation.

The simulation results of the pipeline circuit using the proposed cell library compared favourably in terms of accuracy with respect to 5000 sample Monte Carlo simulations.

The error in the mean and standard deviation predictions for propagation delay were less than 2% and 5% respectively, although in the later case it was typically less than 3%. Regarding the leakage power the maximum error of the mean value was just over 3% but typically below 1%, the error in standard deviation was in general less than 3%.

Blocks	**Number of Gates**	**Computation Time (Days: Hours: Minutes: Seconds)**			**Speed-up Factor compared with**	
		Cell Library	**MC**	**SSTA/SPA**	**MC**	**SSTA/SPA**
Muller C Element	*1*	*0.34 s*	*15m: 29s*	*0.34 s*	*2,732*	*1*
D-flip-flop	*6*	*0.38 s*	*40m: 23s*	*3.25 s*	*6,376*	*8.5*
Toggle Element	*13*	*0.49 s*	*58m: 50s*	*6.67 s*	*7,204*	*13.6*
3-to-8 Decoder	*21*	*0.80 s*	*2h 7m: 14s*	*11.8 s*	*9,543*	*14.8*
Pipeline Reg	*46*	*0.55 s*	*4h: 46m: 26s*	*7.01 s*	*31,247*	*12.7*
Delay Element	*50*	*0.33 s*	*2h: 54m: 59s*	*15.67s*	*31,815*	*47.5*
16-to-1 Multiplexer	*60*	*0.99 s*	*6h: 24m: 21s*	*21.45 s*	*23,294*	*21.7*
Parallel Adder	*178*	*1.79 s*	*1d: 10h: 29m*	*25.4 s*	*69,352*	*14.2*
ALU	*578*	*2.50 s*	*3d: 4h: 8m*	*33.9 s*	*109,632*	*13.56*
Register File	*2370*	*0.96 s*	*13d: 0h: 30m*	*98.7s*	*1,171,875*	*102.8*
Pipeline Circuit	*3272*	*3.27 s*	*33d: 15h: 49m*	*166.1 s*	*889,340*	*50.8*

Table 5-5 Computational time comparison.

Furthermore, the proposed cell library is also much more efficient compared with MC simulation in analyzing the process variation effects on circuit performance; it would take a month to run a 5000 sampled MC simulation for the pipeline circuit whereas the cell library only requires a few seconds. Table 5-5 lists the computation time comparison results for both the cell library and Monte Carlo PDF generation techniques.

Additionally Table 5-5 also compares the CPU time between the cell library and traditional flatten SSTA and SPA approaches; it shows that using the cell library is also much faster in computing the delay and leakage power distribution of circuits over traditional SSTA and SPA approaches. The speed-up factor is highly related to the regularity of the circuits. For example, the computation time for the performance analysis of the Register File (RF) block is at least 100 times faster than SSTA and SPA; this is because there are a large number of identical digital blocks (registers) in the circuit of the RF which can be represented by a single model block. On the other hand the speed-up factor for the decoder block is only around 10 since most of the decoder circuit is modelled at gate level. The experimental results show that the overall speed-up factor for the whole demonstration pipeline circuit is more than 50. On the other hand, since the circuit blocks in the library are characterised using the SSTA and SPA based sensitivity analysis technique, the accuracy of the analysis of process variation effects using flattened SSTA and SPA should be the same as the analysis using the cell library, which is shown in Tables 5-3 and 5-4. Figures 5-32 to 5-34 show the graphic views of the analytical experimental results for the main blocks using in the pipeline circuit, including the error analysis and computational time comparison.

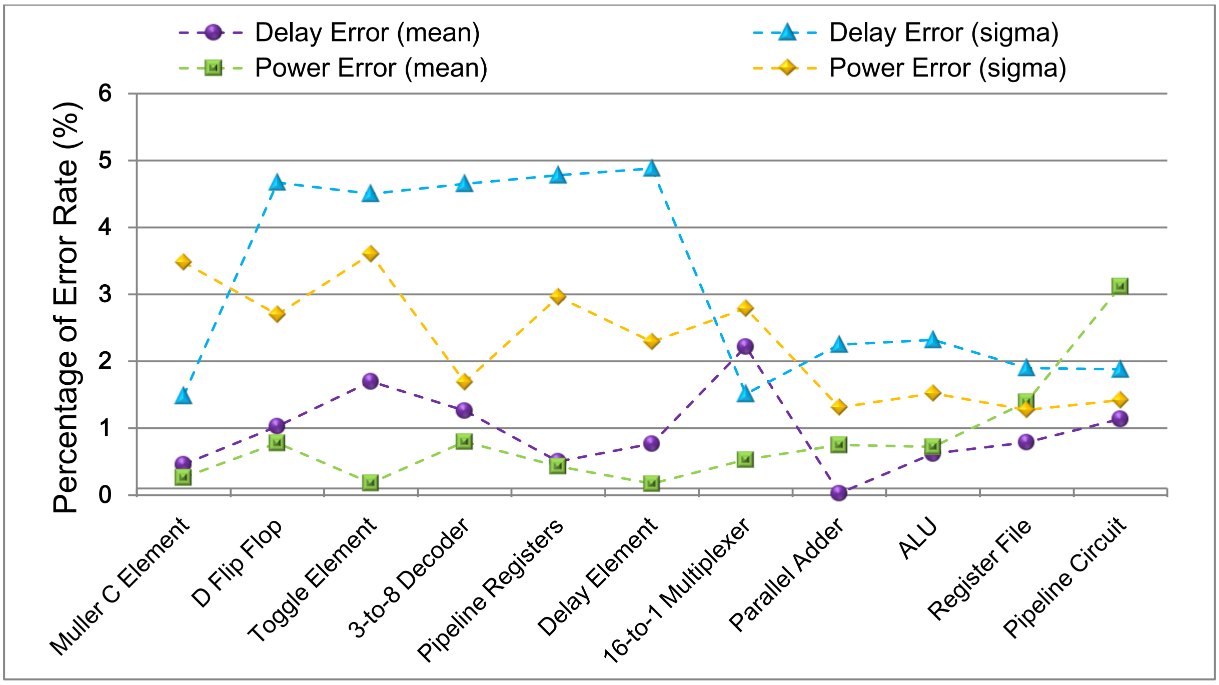


Figure 5-32 Error analysis of the experimental results.

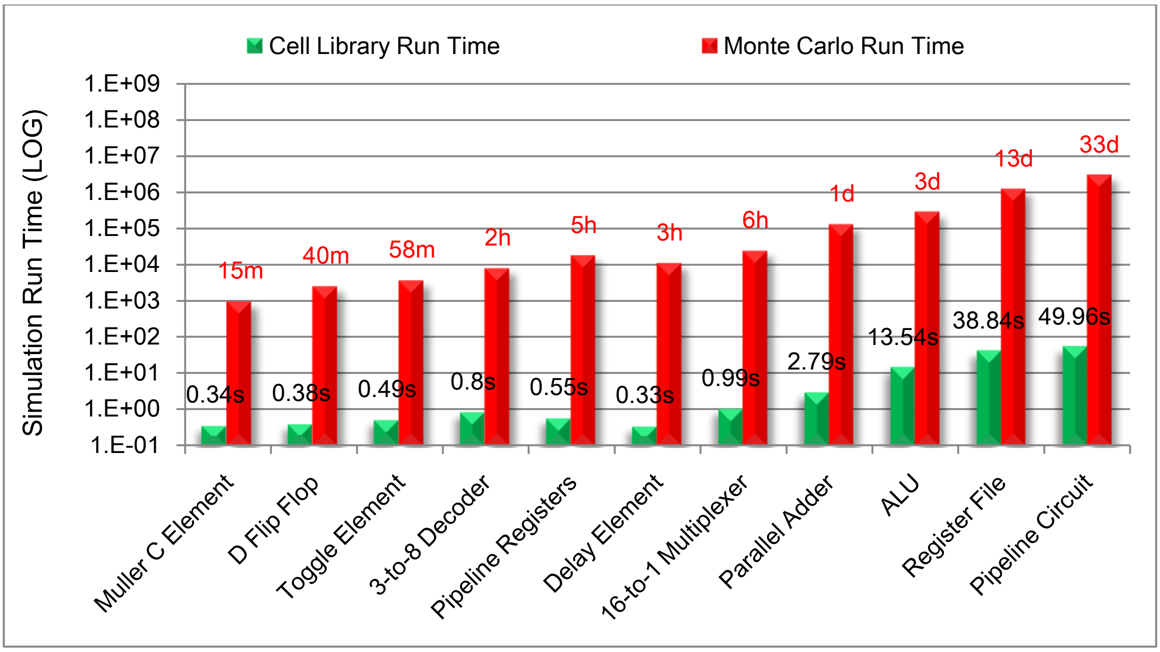


Figure 5-33 Computational speed compared with MC simulation.

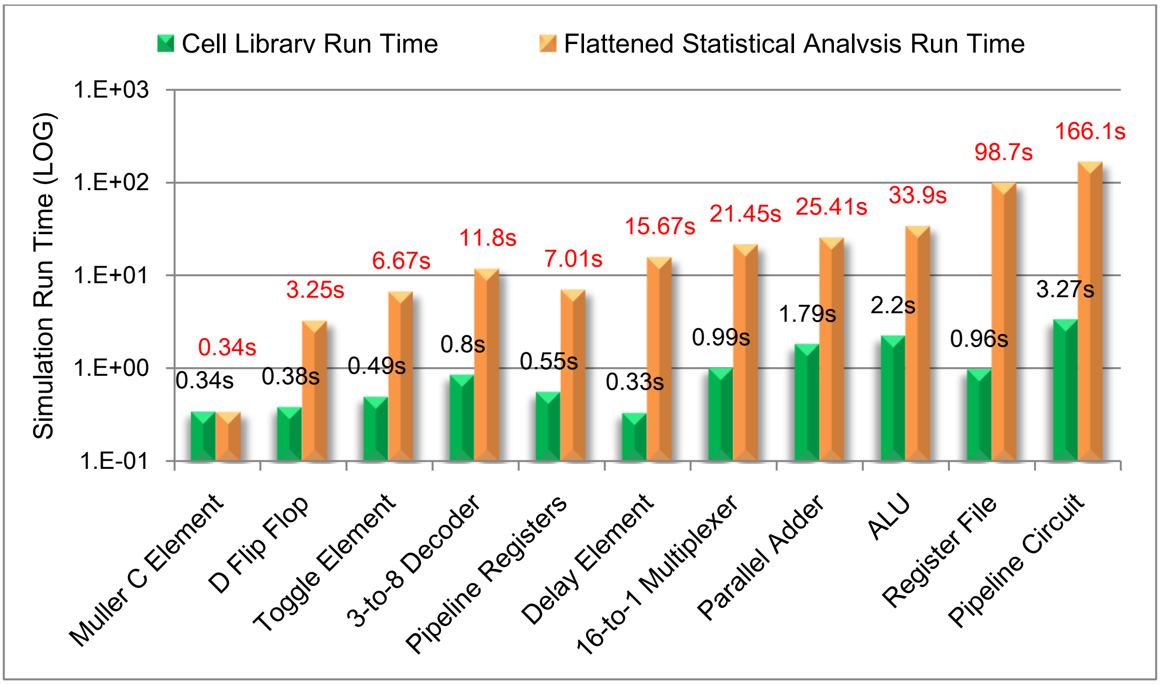


Figure 5-34 Computational speed compared with flattened statistical analysis.

Since the circuit size of some blocks used in the pipeline circuit (*ALU*, *Register File* and *Whole Pipeline Circuit*) are relatively large, it takes several days (in the case of the pipeline circuit approximately a month) to run the 5000-sample Monte Carlo (MC) simulation. It is not feasible to do the experiment with such a long computation time; there is also a probability that such a long SPICE simulation will crash leading to the incomplete result data. Consequently the actual Monte Carlo delay data for the last three blocks in Table 5-3 has obtained by running the simulation only for the signal paths with signal transitions, instead of simulating the whole circuit wherein most of the gates are inactive. On the other hand, the Monte Carlo runs for leakage power are much faster since only static simulation is required. All of the MC simulation results for leakage power shown in Table 5-4 were completed within a day. The computation times listed in Table 5-5 is the time needed for both delay and leakage power simulations. The CPU times for the larger blocks in Table 5-5 are actually predicted by timing the first 50 simulation runs, and then multiplying the resulting CPU time by 100; these predicted times are probably underestimated since the MC simulations will usually suffer a speed deceleration during the computation process.

5.5 Summary

The implementation of the cell library and experimental results of a demonstration pipeline circuit constructed using the library blocks are outlined in this chapter. The methodology to characterise higher level blocks has been described first. All the commonly used standard cells are constructed using SPICE based sensitivity analysis and each cell provides the statistical analysis protocol. In order to improve the computational efficiency of the cell library, the higher level blocks can be characterised using SSTA and SPA results using the existing circuit blocks in the library.

All the library cells are implemented in MATLAB SIMULINK, which provides a friendly graphic user interface. Any desired circuit can be built up using the cell library blocks and the process variation effects on its delay and leakage power performance can be analyzed. After a initialization process which defines the variational sources and their range of variations, and a pre-simulation setting which setups circuit input and load conditions, the circuit simulation can be performed in a SIMULINK environment. All the simulation results, including the functional waveform and performance parameter distributions, can be plotted using the corresponding MATLAB functions in the cell library tool set.

A full analysis has been demonstrated on a 2-stage micropipeline circuit; where it has been shown that this technique can achieve an accuracy comparable to that obtained from a Monte Carlo simulation with the errors less than 5%, as well as saving a significant amount of computation time. On the other hand, the analysis speed of the proposed cell library is also relatively faster than the traditional flattened SSTA and SPA techniques, the speed up factors for the main blocks using in the demonstration pipeline circuit are ranged from 10 to 150 depend on their circuit regularity. Since the cell library using the SSTA and SPA techniques to characterize higher level blocks, the accuracy of the analysis results using both the cell library and fattened statistical approaches should be no difference.

5.6 Reference

[1] M. website. *MATLAB overview*. Available: http://www.mathworks.co.uk/products/matlab/?s_cid=wiki_matlab_15

[2] MathWorks. *Simulink product page*. Available: http://www.mathworks.co.uk/products/simulink/?s_cid=wiki_simulink_8

[3] I. E. Sutherland, "Micropipelines," *Communications of ACM*, vol. 32, pp. 720-738, 1989.

[4] S.-M. Kang and Y. Leblebici, *CMOS Digital Integrated Circuits: Analysis and Design*, 2nd ed.: McGraw Hill Higher Education, 1999.

[5] S. Yan, *Fundamentals of digital electronic technology*, 5 ed.: Higher Education Press, 2003.

[6] R. E. Miller, "Sequential circuits," *Switching Theory*, vol. 2, 1965.

CHAPTER 6

CONCLUSION AND FUTURE WORK

This concluding Chapter summarizes the salient points presented in this thesis and highlights important conclusions. This is followed by few key points for future work.

6.1 Summary and Conclusions

Variability in the delay and leakage power consumption of CMOS devices, circuit and systems arises from scaling VLSI circuit technologies beyond the ability to control specific speed-dependent and power-dependent parameters. This erosion in device parameter values, as well as the already-critical problem of the environmental parameter uncertainty, has elevated variability to a major limitation to continued technology scaling. Attempts to improve target parameter values in the manufacturing process are now confronted by atomistic-level constraints, which make the achievement of target values extremely difficult, especially at the nanometer technology node. Therefore, the circuit delay and leakage power performance have to face the accuracy problem, which makes these parameter values deviate in a certain range with respect to their nominal values, caused by the process variation effects introduced during fabrication and the operating environment. Consequently, the performance parameters, such as delay and leakage power dissipation, of present day circuit designs, must be evaluated before fabrication in order to predict any possible yield loss due to the process variation effects.

An architectural level modeling methodology for circuit propagation delay and leakage power dissipation prediction is proposed in this thesis. A statistical cell library has been built in order to provide both speed and efficiency in analyzing circuit delay and leakage power performance. The thesis started with a review of the background on semiconductor technology scaling and the impact of the consequent process variation effects on device and circuit performance. The different source and components of variations were also discussed in this chapter. Generally speaking, there are two main types of variaitonal sources which significantly affect the circuit performance. One is manufacturing process variation which involves a wide range of process parameters during fabrication, such as threshold voltage adjustment implantation energy, High-k dielectric thickness, substrate doping etc. These parameters cannot be easily controlled as fixed values, and will cause the fluctuations in terms of the manufactured devices parameters thus having a permanent affect on the circuit design. The other type of variational sources comes from the circuit operating environment, such as the supply voltage and temperature, which will cause temporary variations in circuit performance parameters. Each variational source can be divided into two components, inter-die and intra-die variations, which affect the circuit performance in different ways. The inter-die variation refers to the difference in variation effects between lots, wafers and dies. On the other hand, the intra-die variation captures the independent variation effects within each die. Furthermore, since the semiconductor technology has merged into the nanometer range, the intra-die variations become more and more significant and can no longer be neglected. In order to model and analyze the process variation effects on the present day circuit performance, both variation components must be taken into consideration.

In Chapter 2, the different methodologies and approaches used for modeling process variability has been reviewed. The variability aware analysis can be roughly divided into 2 steps: process-to-device and device-to-circuit analysis. The process-to-device analysis is used to abstract effects of the process parameter variations on the device physical parameter values, which is commonly achieved using TCAD tool based on DoE and RSM techniques. The work in this thesis is based on the second class of analysis, device-to-circuit variability analysis, and extended to the architectural level. The variation information on device parameters are assumed to be given and the environmental variation sources are also considered in this work. The effects of these variation sources on circuit performance parameter, such as delay and leakage power dissipation, will be estimated after the analysis.

The traditional variability aware analysis approaches are based on the worst case corner model, in which the device parameter variations are represented using its best and worst corners. Therefore the corresponding circuit parameters are also modeled using their performance boundaries after the Worst Case Analysis (WCA). This modeling technique provides a low computational complexity and rapid analysis speed, so it has been widely used in the industry to evaluate the circuit performance in order to meet the design specifications. However, it confronts some major limitations which have been especially exacerbated in nanometer technology. The main drawback of WCA is that it considers the circuits perform under conditions (best and worst) which rarely happen. Therefore the analysis results of WCA will be either pessimistic or optimistic. Furthermore, the WCA result only provides the variation range of the circuit performance parameters, but no information about how they are distributed in this range. On the other hand, the WCA completely ignores the intra-die variations. In order to meet all these drawbacks of WCA, the variability analysis techniques have been developed towards the statistical domain during the last decade, which characterize the circuit performance parameter variations as probability distributions.

The simplest and most accurate statistical analysis technique for the process variation effects on circuit performance parameters is the Monte Carlo approaches, which performs random sampling repeatedly in a finite space (variational device parameters) and constructs the corresponding probability distribution based on the samples. However, the MC approaches are computational very costly in order to maintain a reasonable analysis accuracy. It requires numerous simulation runs to sample the random values of circuit performance parameters and becomes unfeasible to perform on a very large system. Therefore, the MC techniques are usually used as the reference results to verify other statistical analysis techniques and not as an efficient solution to analyze process variation effects on a very large circuit.

The other statistical approaches, such as Statistical Static Timing Analysis (SSTA) and Statistical Power Analysis (SPA) have been developed extensively during the last decade, where the variational parameters are modeled as Gaussian variables. The gate delay and power distributions are represented as low-order polynomials of these variable parameters, and the circuit delay and power distributions will be computed from these individual gate models. However, the analysis at such a low level (gate level) of design abstraction is inefficient and error prone because the present day circuit designs are very large. The models for more complicated circuit blocks, such as register, FIFOs and ALUs etc., should be available in order to analyze the process variation effects on circuit performance at a higher level of design abstraction. The proposed cell library characterization methodology in this thesis can perform an architectural level analysis of the process variation effects on the delay and leakage power distributions. The details of how to characterize the standard gate cells are described in Chapters 3 and 4.

A first-order canonical delay model was employed to characterize the process variation effects for a single cell in the statistical cell library which takes both the inter-die and intra-die variations into consideration. The signal arrival time at each input node and the delay time of each cell in a circuit are represented as a first-order polynomial of multiple normal RVs with respect to the variation sources. The leakage power of each cell is also modelled as a canonical variable but in logarithmic form because of the exponential relationship between the process parameters and leakage current of a device. The higher-order modelling approach can provide better accuracy when the variations grow larger (over 30% of the mean value); however, the high computation complexity and large amount of fitting experiments required makes it too difficult to implement in a very large circuit. Furthermore, in reality, there are only a small number of process parameters which vary over 30% of their mean values, the typical delay and power PDF of a cell is much closer to a normal distribution. Consequently the first-order polynomial is sufficiently accurate and computational feasible to characterize the process variation effects.

The cell delay PDFs are not only dependant on the process parameters, but also on many other factors such as the slope of the input signal, load capacitance and different input conditions. The statistical cell library takes all of these factors into account by characterizing a cell using a table look-up approach. The different delay values together with the corresponding coefficients with respect to a range of input signal slopes, T_{in}, and output load capacitances, C_{load}, of a basic cell were modelled using a three-dimensional look up table. Each cell requires several independent tables to cope with all the input conditions and output transition cases. In order to reduce the number of SPICE simulation runs to generate each look-up table, a piecewise linear fitting approach was employed in which only 7 typical values are sampled for T_{in} and C_{load} respectively. On the other hand, the leakage power distributions of a gate due to the process variation effects only differ with its static states. Therefore, the LUT for storing the gate leakage power models is much smaller than the one for storing delay models. A two-dimensional LUT can handle all the possible gate leakage power distribution cases.

All the standard cells in the library have been built with the SSTA and SPA protocols in order to propagate their delay and leakage power models through the circuit. The tightness probability based timing analysis technique and the *Wilkinson's approximation* based recursive power analysis approach is employed. The key idea of both the delay and power analysis methodologies is to re-express the non-normal internal operation results into the normal canonical form by matching the first two moments with the analytical results. This will keep the SSTA and SPA alive in the circuits. With the help of the moments matching based statistical timing and power analysis techniques, any desired circuit can be constructed using the standard cells in the library and the impact of process variation effects on the circuit delay and leakage power performance can be analyzed.

The methodology to characterize the higher-level functions blocks, such as registers, decoders, multiplexers, FIFOs, ALUs etc., is outlined in Chapter 5. The SPICE simulation based sensitivity analysis, which is used to characterize the standard cells, is no longer required. Since all the necessary gate cells are already available in the library, the higher-level blocks can be characterized using the SSTA and SPA results of the existing cells in a hierarchical manner. Once a new block has been constructed, it can be further used to characterize other larger blocks. Since only the top level blocks are used for the next-level circuit characterization, the characterized block provides a much faster analysis speed than the flattened SSTA and SPA using the standard cells. The more complicated circuit blocks with a larger number of inputs may require a massive memory space to store the LUTs for all possible switching cases. However, it can be optimised to a variable degree according to the regularity of the circuit. With the proposed characterization approach, the cell library has been expanded to comprise most of the commonly-used circuit blocks with the complexity ranging from a single gate to more than 3272 gates, which are listed in Table 5-1 of Chapter 5.

The whole cell library is implemented in MATLAB SIMULINK. The simulation process of the circuit constructed using the library cells are introduced in the second section of Chapter 5. It starts with an initialization process for choosing the desired global and local variation sources and setting the corresponding variation ranges. There are 5 process and environmental parameters available to be chosen: the supply voltage, operating temperature, effective channel length, the NMOS threshold voltage and PMOS threshold voltage. Since the sensitivities of parameters are derived independently, it is simple to add any other desired parameters as variational sources into the cell library tools if necessary. After the setup of the input signal and load capacitance, the constructed circuit using the library cells can be simulated in SIMULINK. The resulting delay and leakage power PDFs of the circuit under simulation can be plotted using the corresponding MATLAB functions in the cell library tool set.

To demonstrate the feasibility of this modeling approach, a variety of logic blocks of complexities ranging from a single gate to several thousand gates were analyzed with respect to the effects of process variation on propagation delay and leakage power consumption. The experimental results are shown in the Section 5.3 of Chapter 5.The technique compared favorably in terms of accuracy with respect to 5000 sample Monte Carlo simulations. The error in the mean and standard deviation predictions for propagation delay were less than 2% and 5% respectively, although in the later case it was typically less than 3%; regarding the leakage power the maximum error of the mean value was just over 3% but typically below 1%, the error in standard deviation was in general less than 3%. Furthermore the proposed cell library can save a huge amount of computational time compared to the MC simulations with the speed-up factor ranging from several thousands to one million depending on the circuit size. On the other hand, the experimental results using the cell library have also been compared with the traditional flatten SSTA and SPA techniques. It has been shown that the cell library offers a rapid analysis of process variation effects on circuit delay and leakage power dissipation with up to 150 times faster computational speed for the large blocks.

Based on the experimental results, the delay errors of the circuits involving sequential elements, such as flip flops, toggle element and pipeline register etc., are usually larger than the combinational circuit blocks with a similar size. The signal flows in sequential circuits are typically complex because of the existence of feedback signals from their outputs. Therefore, the delay models for these sequential blocks are characterized directly from the SPICE simulation based sensitivity analysis instead of the SSTA based analysis using the existing gate cells in the library. However, the delay distribution of the characterized blocks still show slightly larger errors than the combinational logic blocks since their actual delay characteristics deviate away from the normal models. These modeling errors are not very significant (still within 5% for most cases), and their corresponding effect on analysis accuracy will decrease when the circuit size grows larger and the contribution of the sequential element to the total path delay becomes smaller. On the other hand, the experimental results also show the leakage power errors of the blocks decrease with the growth of their circuit size. This is because the static power distributions of small circuits are very wide due to the exponential relationship between the leakage current and process parameter values. These distributions are closed to lognormal form, and the small errors during the sensitivity analysis will be emphasized after the logarithm transfer. However, as discussed in Chapter 4, the leakage power distributions of a circuit will develop towards to the normal distribution with the increase in circuit size. Therefore, the larger blocks show smaller leakage power modeling errors.

As described in Chapter 5, the demonstration circuit has been analyzed using three techniques: the proposed cell library, Monte Carlo simulation and the fattened statistical approach for the comparison of their computational times. (The MC simulation results are also used for the purposes of accuracy verification). The MC simulations are computationally very costly, hence not feasible to evaluate a very large circuit. It may take weeks to analyze the demonstration pipeline circuit. Both the cell library and the traditional flatten statistical approach (SSTA and SPA) can save a significant amount of computational time compared to the MC technique. Since these two methodologies use the same mathematical algorithms, the corresponding analysis accuracies are similar. However, since the cell library characterized the library circuit blocks in a hierarchical manner, it offers faster computational speed compared with the flattened analysis. The run time for the SSTA and SPA grows with the increase of circuit size, which still takes a few minutes to analyze the whole pipeline circuit. If the circuit under analysis, for example is a large processor system, it may cost hours to run SSTA and SPA on it. The analysis using the cell library, on the other hand, remains a steady low speed since the internal circuit of an existing block requires no SSTA and SPA computation, instead using the pre-characterized LUTs. The analysis speeds of the functional blocks in the library depend on the regularity of the actual circuits. Highly regular circuits (more identical circuit segments) require a small number of LUTs to characterize its delay and leakage power distributions, and the simulation requires less memory load time thus shortening the overall computational time, vice versa. For example, the Register File (RF) circuit contains a large number of identical signal paths with the same delay characteristics, so they can be represented by one model which saves a large amount of memory space so speeds up the analysis process. As listed in Table 5-5 in Chapter 5, the speed of the RF block (2370 gates) is more than twice faster than the ALU block (578 gates), whose internal LUTs cannot be optimized much due to the lack of regularity in the circuit. Furthermore, the speed for estimating the circuit delay and leakage power distributions also depends on how the circuit is constructed. If the desired circuit is built up only using basic gate cells, the analysis becomes the flattened SSTA and SPA, which gives up the computational efficiency of the cell library. Therefore, keeping the design abstraction level of blocks used in a circuit as high as possible will speed up the whole analysis of the effects of process variation on circuit delay and leakage power dissipation as efficient as possible.

It is considered that the technique outlined in this thesis, permits designers to efficiently assess the effects of variations in processing parameters, such as effective gate length and threshold voltage, together with supply voltage, on a design in terms of their potential impact on specification parameters such as propagation delay and leakage power early in the design cycle. The circuit under analysis can be constructed using higher level blocks instead of using basic gate cells. The constructed circuit can be characterized as a new block and saved back in the library for further expansion. On the other hand, as discussed previously, the cell library is portable for additional process parameters, which can be simply achieved by loading in the delay and leakage power sensitivity LUTs of the new parameters during the initialization process. Subsequently, the designer can choose which technology or cell library should be used to implement the design, for a given application, to ensure its robustness to the effects of process variation.

The propose cell library tool set not only provides a good tradeoff between accuracy and computation speed in estimating the process variation effects on circuit delay and leakage power performance, but also offers a degree of flexibility which allows users to run the circuit analysis at a architectural level. The methodology presented in this thesis, in general, can be applied to any process technology and capable of adopting any device and environmental parameters whose variation effects will significantly affect the circuit delay and leakage power performance. Moreover, the method can be very useful for optimizing circuit design to achieve better performance and higher yield.

6.2 Future Work

The methods or techniques for variability modeling and analysis will continue to be an important area of research in future technologies. The following section highlights the key points for future work in this area.

- The cell library presented in this thesis is mainly focused on the delay and leakage power modeling, but does not take the dynamic power into consideration because the high characterization complexity. More efficient dynamic power models need to be proposed so that they can be used in the cell library to fill the gap in dynamic power analysis.

- The work presented in this thesis is based on the first-order canonical delay and leakage power model. However, if the device parameter variations are larger than 30% of their mean values, the proposed cell library can still run the simulation and estimate the circuit delay and leakage power distributions but the accuracy of the results is not guaranteed. Higher-order models can provide better accuracy but accruing the penalty of high exponential complexity with respect to the number of process parameters under analysis. The tradeoff problem between the analysis accuracy and complexity of process variation effects on circuit performance parameters due to its importance will be addressed by other researches, so that it may lose the limitation of the proposed cell library to cope with larger parameter variations.

- All the parameter variations in the proposed cell library are assumed to be Gaussian distributions. However, the real distributions of these process parameters are very difficult to estimate, and they could be non-normally distributed. Therefore, the cell library models may take other types of parameter distributions, such as uniform and Poisson distribution etc., into consideration in the future work.

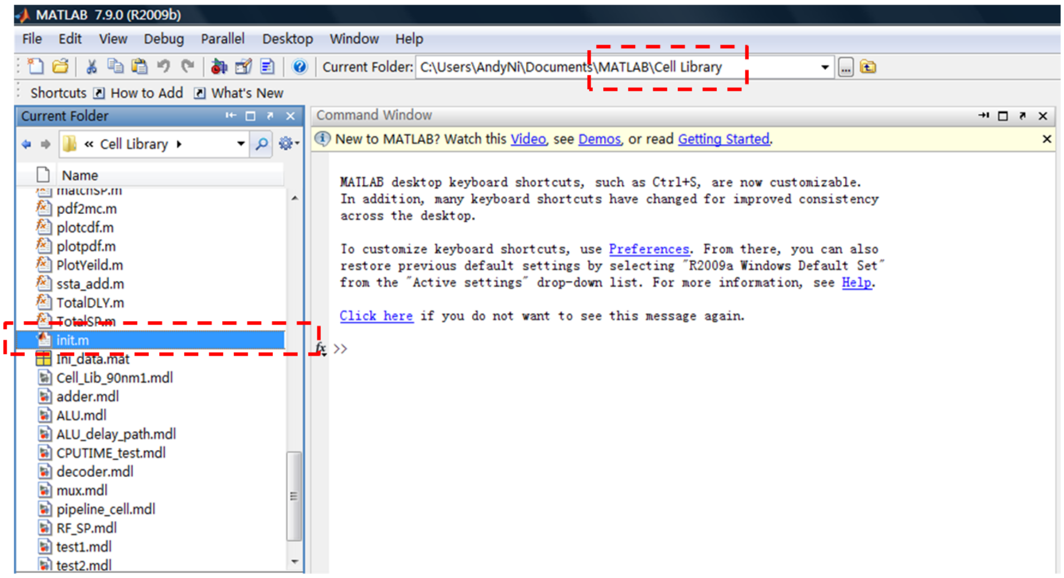
- The proposed cell library does not take the interconnect between devices into consideration, where the delay times are assumed to be zero. The proper interconnect delay model and analysis technique is needed in order to make the cell library robust in evaluating circuit performance due to the process variation effects.

APPENDIX A

USER GUIDE OF THE CELL LIBRARY TOOL SET

A.1 Initialization function "init"

Before using the cell library, there are several settings need to be initialized. In this subsection, the use of the initialization function for predefining process variation specifications will be introduced.

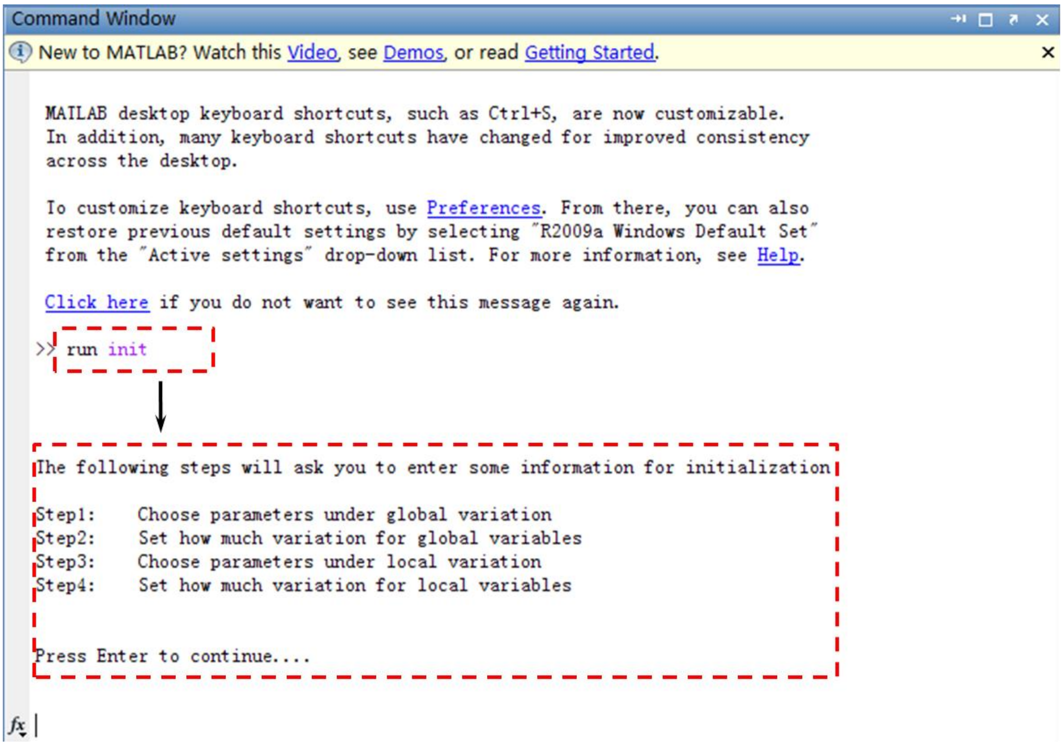


New to MATLAB? Watch this Video, see Demos, or read Getting Started.

```
MATLAB desktop keyboard shortcuts, such as Ctrl+S, are now customizable.
In addition, many keyboard shortcuts have changed for improved consistency
across the desktop.

To customize keyboard shortcuts, use Preferences. From there, you can also
restore previous default settings by selecting "R2009a Windows Default Set"
from the "Active settings" drop-down list. For more information, see Help.

Click here if you do not want to see this message again.
>>
```

Figure A-1 Location of the "init.m" file.

If the current folder of MATLAB is browsed to the proposed cell library tools, an m-file can be found in the "current folder" window called "init.m" as shown in Figure A-1. In order to run this program, typing "*run init*" in the command window, the introduction message will appear which shows the main steps for the cell library initialization as shown in Figure A-2.

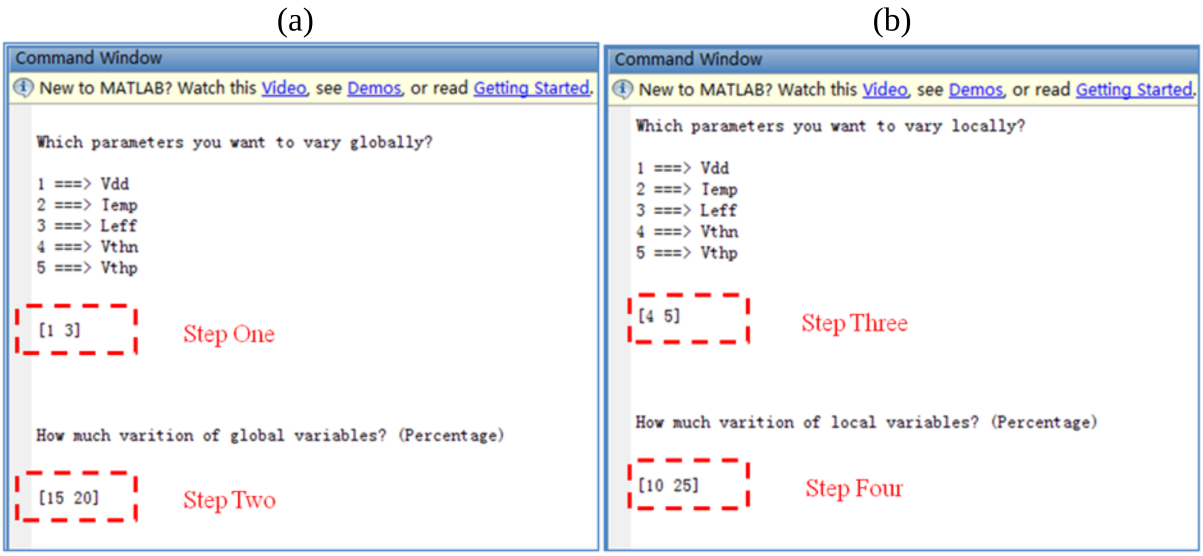


New to MATLAB? Watch this Video, see Demos, or read Getting Started.

```
  MATLAB desktop keyboard shortcuts, such as Ctrl+S, are now customizable.
  In addition, many keyboard shortcuts have changed for improved consistency
  across the desktop.

  To customize keyboard shortcuts, use Preferences. From there, you can also
  restore previous default settings by selecting "R2009a Windows Default Set"
  from the "Active settings" drop-down list. For more information, see Help.

  Click here if you do not want to see this message again.
>> run init

The following steps will ask you to enter some information for initialization

Step1:    Choose parameters under global variation
Step2:    Set how much variation for global variables
Step3:    Choose parameters under local variation
Step4:    Set how much variation for local variables


Press Enter to continue....
```

Figure A-2 Running the cell library initialization program.

Basically there are 4 set of parameters need user to define in the initialization program as shown in the above figure. Firstly, it asks which process parameters are selected as the global variation sources. For demonstration purpose, there are 5 device and environmental parameters available in the cell library: supply voltage V_{dd}, operating temperature T, effective transistor channel length L_{eff}, threshold voltage for N-type device V_{thn} and threshold voltage for P-type device V_{thp}. Each parameter has been assigned with a number from 1 to 5. Choosing the desired parameters can be simply done by typing the indicated numbers in a vector form in the command window. Figure A-3 (a) shows an example in where the 1st and 3rd parameters (V_{dd} and L_{eff}) in the menu are selected as global variation sources. The program will ask how much variation specified for each global parameter in the following step. The percentage of the 3 sigma values of the global variation sources can be defined using the same form as the previous step. As shown in Figure A-3 (a), V_{dd} and L_{eff} are set to deviate 15% and 20% to their mean values respectively. The next 2 steps of the initialization program are exactly the same as the first 2 steps, but defining the specification for local variables. Figure A-3 shows an example in where V_{thn} and V_{thp} are selected as the local variation sources and the 3 sigma values are set to 10% and 25% with respect to their mean values respectively.

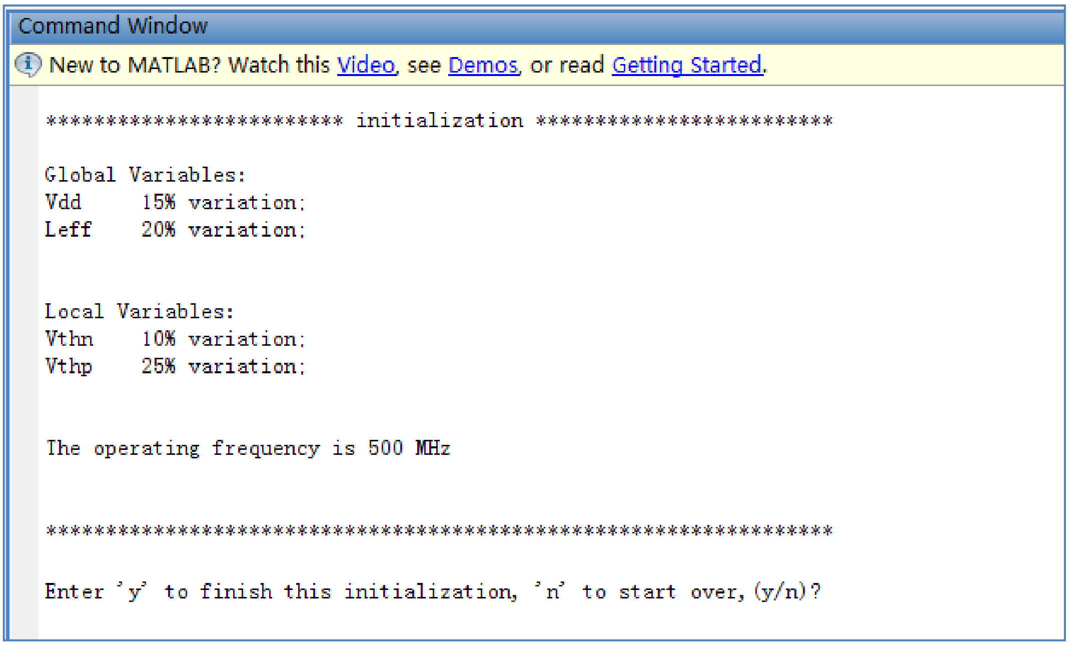

(a)
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Step One
Step Two
(b)
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Step Three
Step Four

```
Which parameters you want to vary globally?

1 ===> Vdd
2 ===> Temp
3 ===> Leff
4 ===> Vthn
5 ===> Vthp

[1 3]


How much varition of global variables? (Percentage)

[15 20]
```

```
Which parameters you want to vary locally?

1 ===> Vdd
2 ===> Temp
3 ===> Leff
4 ===> Vthn
5 ===> Vthp

[4 5]


How much varition of local variables? (Percentage)

[10 25]
```

Figure A-3 (a) Initialization for global variables; (b) Initialization for global variables.

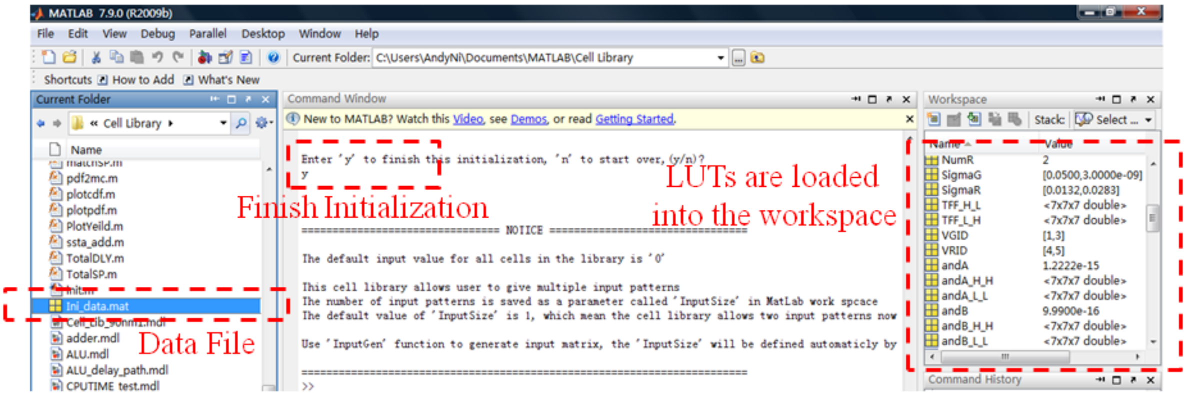
After defining all the variation specifications, a summery sheet will be generated as shown in Figure A-4. That is for the final check of the initialization. If any error happens by mistake when tying the command, the whole process can be go over again by entering "*n*". If nothing is wrong, the initialization can be finished by entering "*y*".


Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.

```
************************ initialization ************************

Global Variables:
Vdd      15% variation;
Leff     20% variation;


Local Variables:
Vthn     10% variation;
Vthp     25% variation;


The operating frequency is 500 MHz


*****************************************************************

Enter 'y' to finish this initialization, 'n' to start over, (y/n)?
```

Figure A-4 Summery sheet for the initialization.

When the initialization is finished, a MATLAB data file "Ini_data.mat", which can be founded in the *current folder* window, will be loaded into workspace automatically as shown in Figure A-5. This file comprises the delay and leakage power LUTs for all the library cells which may be used for the further process variation effects analysis. The data file can be modified by users in cases of adding new cells to the library or changing the semiconductor technology nodes.



```
Enter 'y' to finish this initialization, 'n' to start over,(y/n)?
y

=============================== NOTICE ===============================

The default input value for all cells in the library is '0'

This cell library allows user to give multiple input patterns
The number of input patterns is saved as a parameter called 'InputSize' in MatLab work spcace
The default value of 'InputSize' is 1, which mean the cell library allows two input patterns now

Use 'InputGen' function to generate input matrix, the 'InputSize' will be defined automaticly by

======================================================================
>>
```

Name	Value
NumR	2
SigmaG	[0.0500,3.0000e-09]
SigmaR	[0.0132,0.0283]
TFF_H_L	<7x7x7 double>
TFF_L_H	<7x7x7 double>
VGID	[1,3]
VRID	[4,5]
andA	1.2222e-15
andA_H_H	<7x7x7 double>
andA_L_L	<7x7x7 double>
andB	9.9900e-16
andB_H_H	<7x7x7 double>
andB_L_L	<7x7x7 double>

Figure A-5 Loading LUTs for delay and leakage models of cell library.

Additionally, a notice message will show up in the end of the initialization program, which notices the users that a variable called "*InputSize*" is generated and its default value is 1. This variable together with the "*InputGen*" function are for defining the input signal for a circuit, which will be described in detail in the following subsections.

A.2 Circuits construction using the library cell

The cell library interface can be opened by click the file "*Cell_Lib_90nm.mdl*" in the *current folder* windows. A SIMULINK library interface will pump up as shown in Figure A-6. Since SIMULINK shares the work space of MATLAB, the pre-loaded LUTs for the delay and leakage power models during the initialization can be used for all library cells. As described in the previous subsection, all the user defined parameters are also stored in the MATLAB workspace. Therefore, all these data can be treated globally which lead to a significant memory space saving. Additional, since the computational speed of MATLAB is much faster than SIMULINK, it is more efficient to store LUTs and argument parameters in MATLAB and perform further calculation.

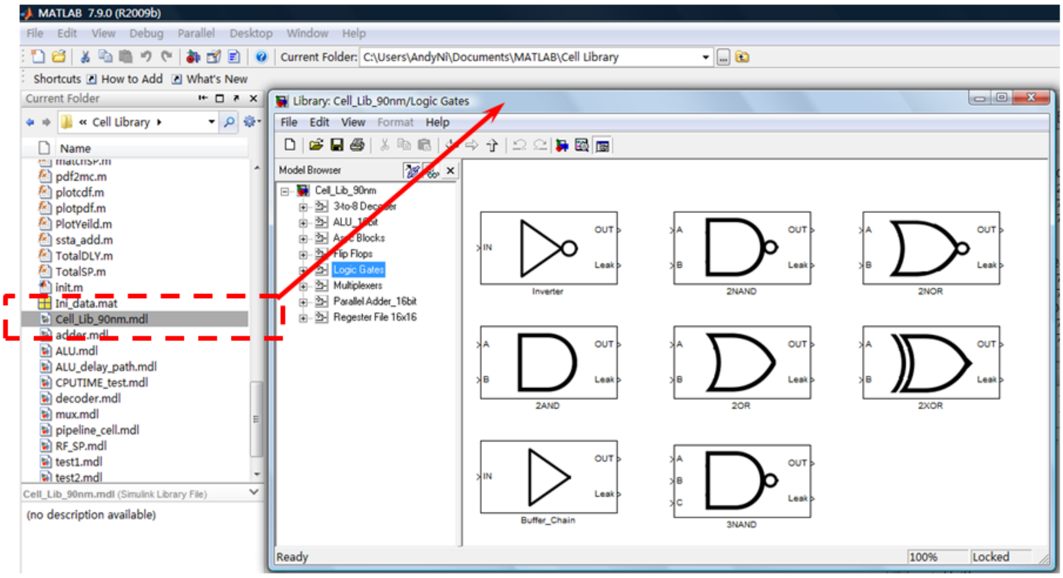


Figure A-6 Cell library in SIMULINK.

Each cell in the library has an extra output pin, *Leak*, which represent the leakage power dissipation. The library cells can be directed dragged from the library widow into a new SIMULINK model file to build up any desired circuit. Figure A-7 shows an example 2-1 multiplexer circuit constructed using the library standard cells.

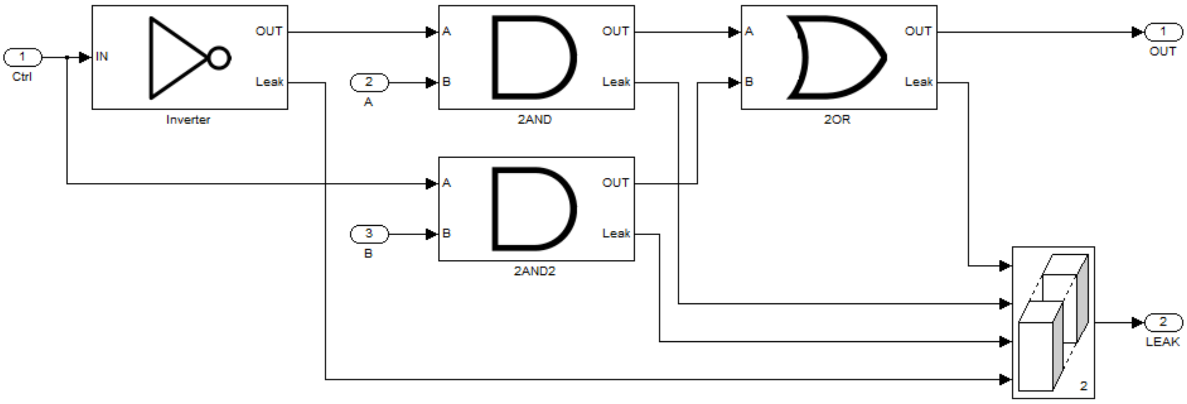
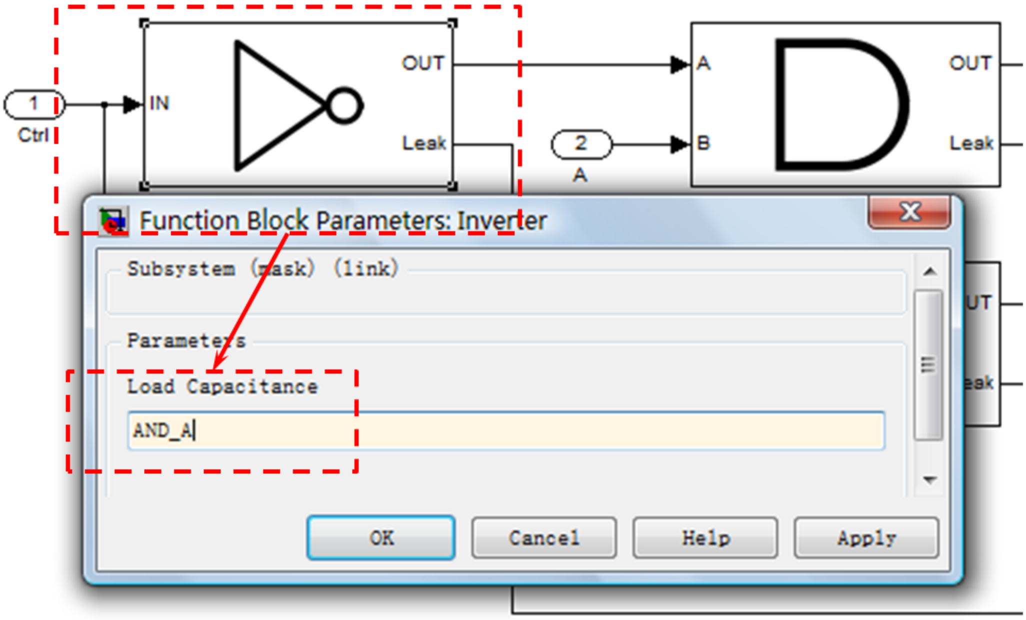


Figure A-7 Constructing a 2-1 multiplexer using the library cells.

As shown in the above figure, the *leak* terminals of all the cells in a circuit need to be connected to a vector combiner for further the calculation of the total circuit leakage power distribution. On the other hand, in order to make the statistical timing analysis result as accurate as possible, the load capacitance of each cell in a circuit needs to be defined. As the load capacitance of a cell is the sum of the input capacitances of the fan-out cells, the the input capacitance values of all input terminals of all cells in the library are also stored in the initialization data file, which have already loaded into the workspace of MATLAB. This makes it easier when setting the load capacitance of each cell. A meaningful constant name can be used rather than input a capacitance value which probably needs to be evaluated in other CAD tools, such as SPICE. Figure A-8 shows an example of how to set the load capacitance of the inverter in the 2-1 multiplexer circuit. "*AND_A*" is the input capacitance value of a AND gate cell at input terminal A, which is stored in the initialization data file and loaded into the workspace. Additionally, the mathematical expression, such as addition, can be directly used to combine multiple fan-out load capacitances when setting the cell load condition in a circuit.



Function Block Parameters: Inverter

Subsystem (mask) (link)

Parameters

Load Capacitance

AND_A

Figure A-8 Load capacitance setting for the inverter in the circuit.

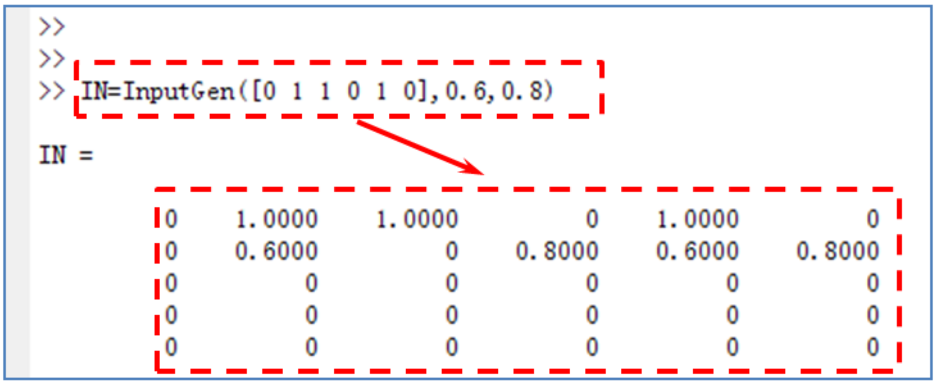
The inputs and outputs of each library cell are not just digital signals, but the data matrixes which contains the signal delay distribution model with respect to the defined process parameter variations during the initialization process. Therefore, the total delay distributions will be indicated from the output terminals of the circuit after simulation.

A.3 *Circuit simulation using the library cell*

The function "*InputGen*" in the cell library tool, which can be located in the *current folder* window, is used for generating the primary input data sequence for the cell library. The function format of "InputGen" is shown in Equation 5.1:

$$\textit{Input Matrix} = \textbf{\textit{InputGen}}\ (\textit{InputData}, \textit{R\_time}, \textit{F\_time})$$

There are 3 input parameters for "*InputGen*", *InputData* represents input digital data sequence which is the first row of the input signal matrix as described before, *R_time* and *F_time* indicate the rising and falling time of the inputs signal respectively. The function will return a signal matrix and load it into workspace, thus is can be used directly in SIMULINK. Since the generated matrix is for the primary input of a circuit, there is no variation exist and the all values below Row 2 of the matrix are zeros. If the input signal of a circuit is generated by "*InputGen*", the parameter *InputSize*, which will be used in further analysis and simulation internally, is automatically set to row length of the input signal matrix. If the input signal can also be created manually in MATLAB, *InputSize* should also be set to an appropriate value manually. Otherwise there will be errors during simulation. Figure A-9 shows an example for using "*InputGen*" to generate a "0 1 1 0 1 0" data sequence with rising transition time equal to 0.6s and falling transition time equal to 0.8s.



```
>>
>>
>> IN=InputGen([0 1 1 0 1 0],0.6,0.8)

IN =

         0    1.0000    1.0000         0    1.0000         0
         0    0.6000         0    0.8000    0.6000    0.8000
         0         0         0         0         0         0
         0         0         0         0         0         0
         0         0         0         0         0         0
```

Figure A-9 Using "*InputGen*" to generate input signal for a circuit under simulation.

After setting up the input stimulus, the circuit can be simulated by clicking the "Start Simulation" button in the upper tool bar of SIMULINK model file as shown in Figure A-10.

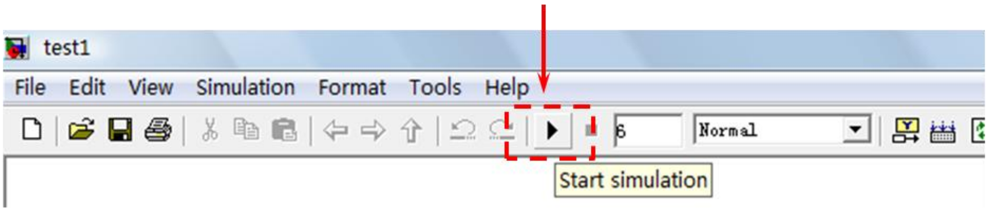


Figure A-10 How to start a circuit simulation.

After simulation, the delay distribution at each output of the circuit and the total leakage power distribution will be loaded to the workspace in data matrix forms.

A.4 Result PDF plotting

The PDF of simulation delay and leakage power results, which already loaded to the workspace, can be plotted using the MATLAB functions in the tool set as below:

*"Plotting Delay PDF" =**PlotPDF** (delay_matrix)*

*"Plotting Leakage Power PDF" =**PlotSP** (SP_matrix)*

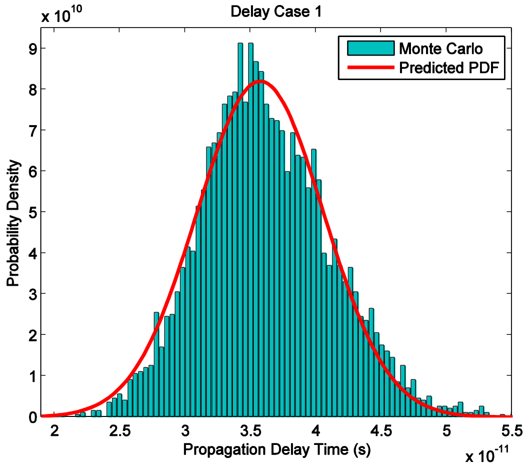
These two functions will not retune any values, the delay and leakage power PDF graphs will be automatically plotted in the new graph windows. The function to generate the histogram of MC data is also available in the cell library tool set as shown below:

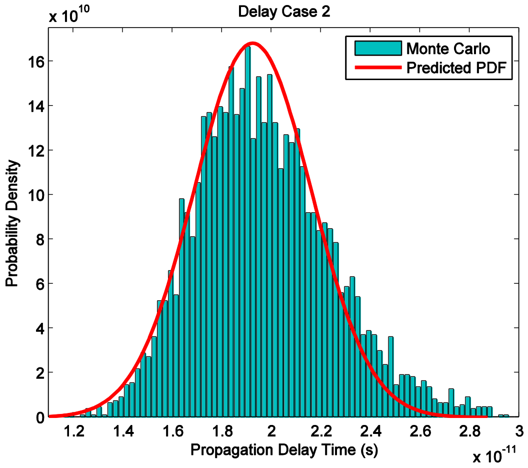
*"Plotting MC histogram" =**HistMC** (MC data)*

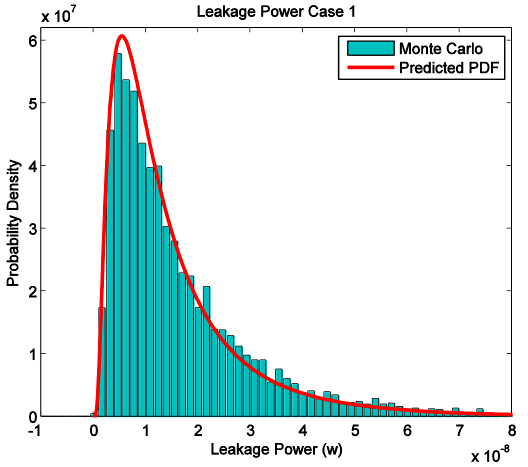
APPENDIX B

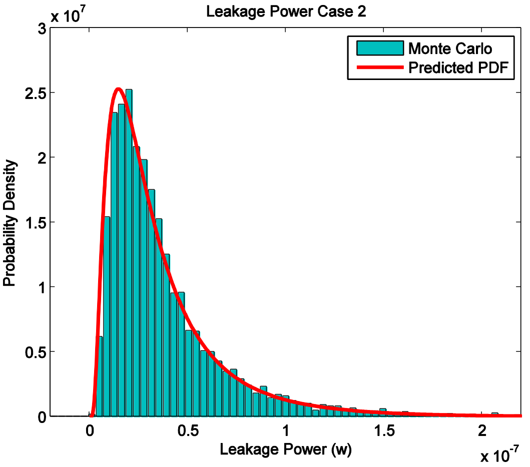
ADDITIONAL EXPERIMENTAL RESULTS

B.1 XOR gate

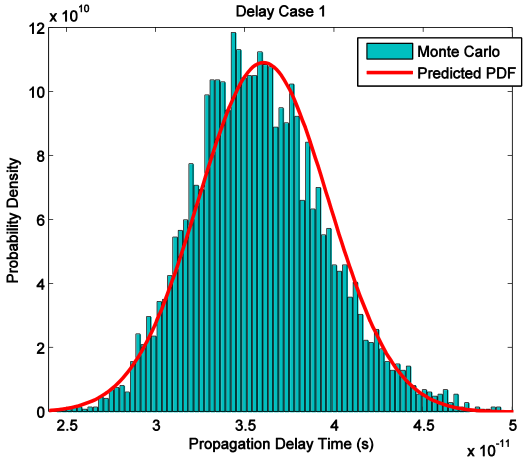

Delay Case 1

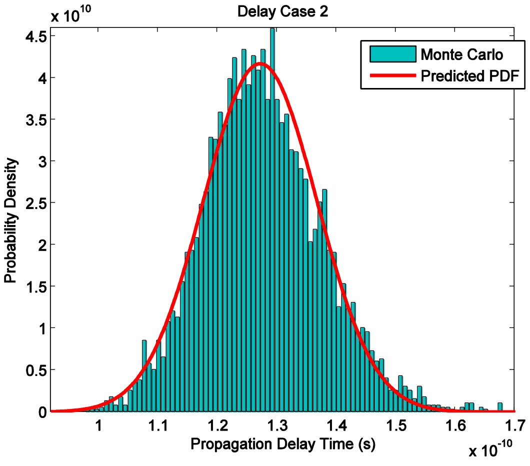

Delay Case 2

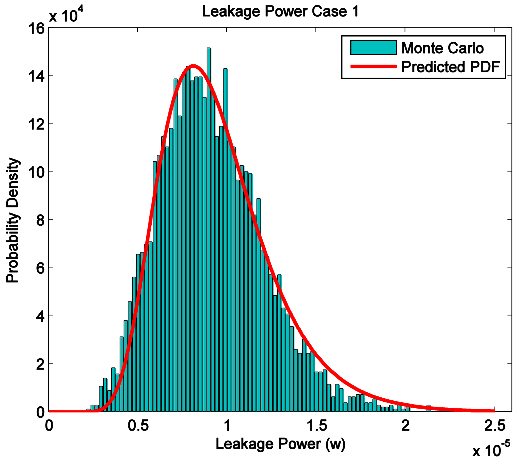

Leakage Power Case 1

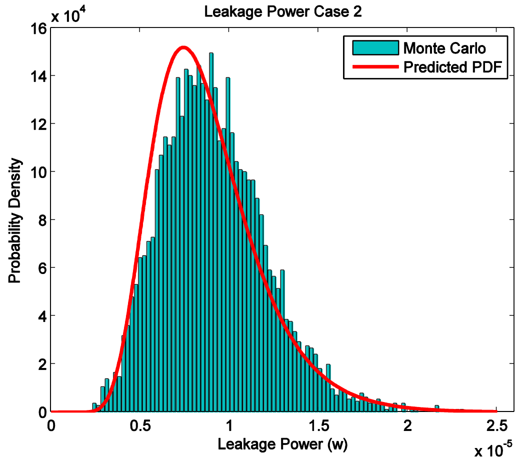

Leakage Power Case 2

B.2 Capture-Pass Flip Flop

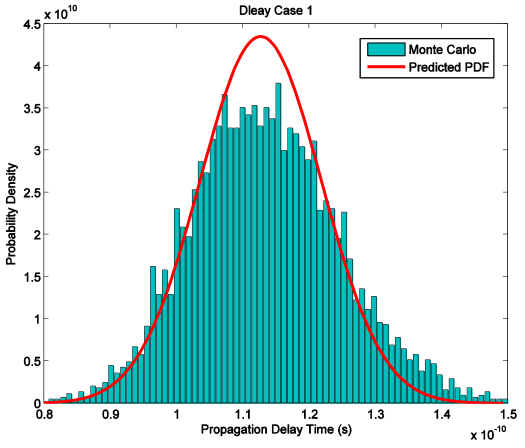


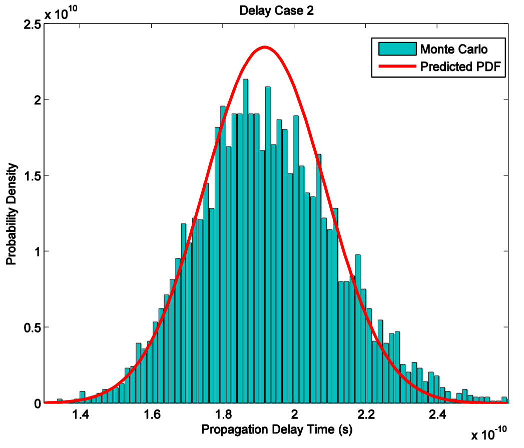


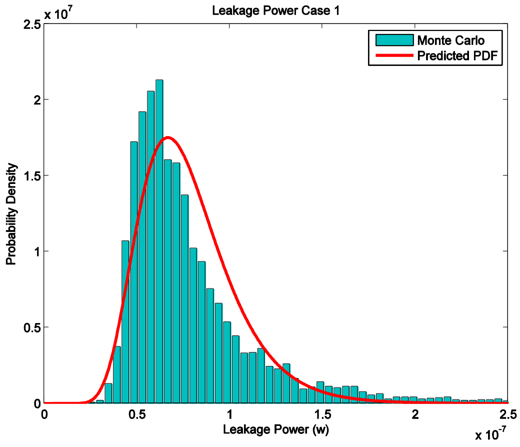


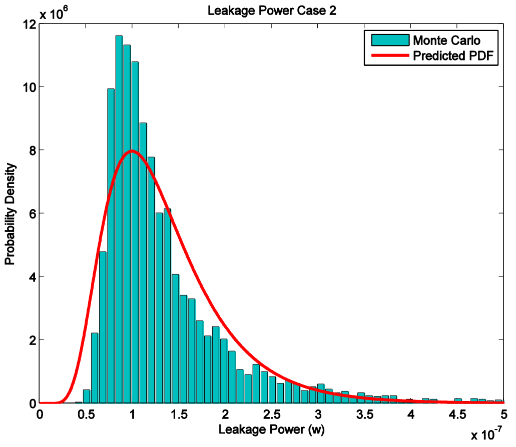


B.3 Toggle Element

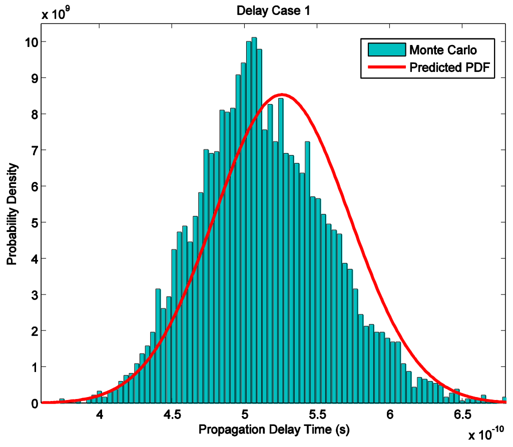


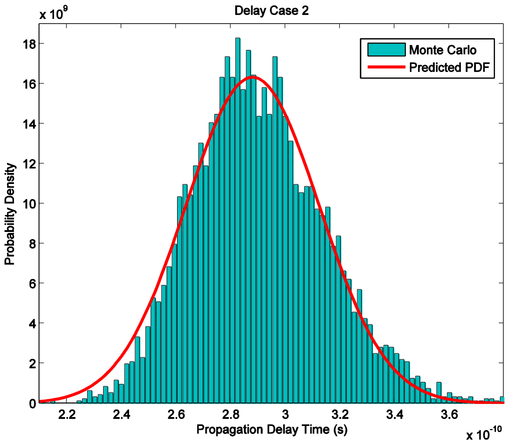


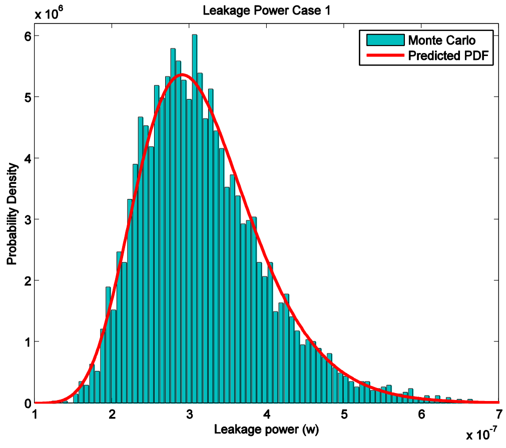


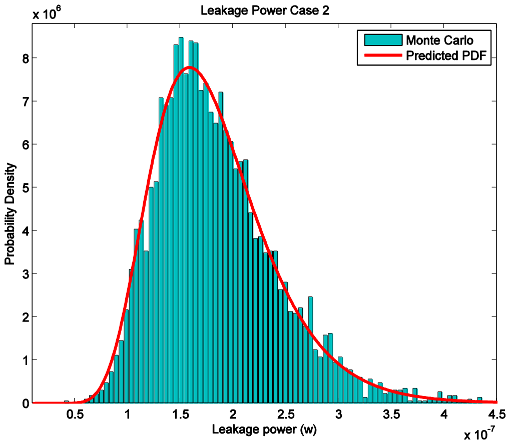


B.4 16-1 Multiplexer

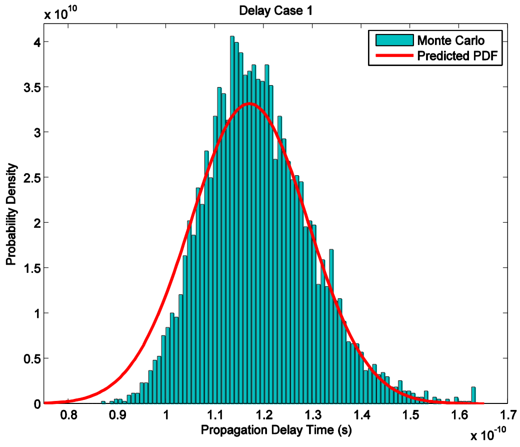


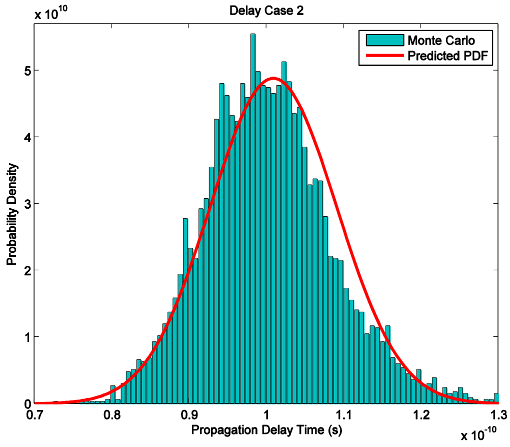


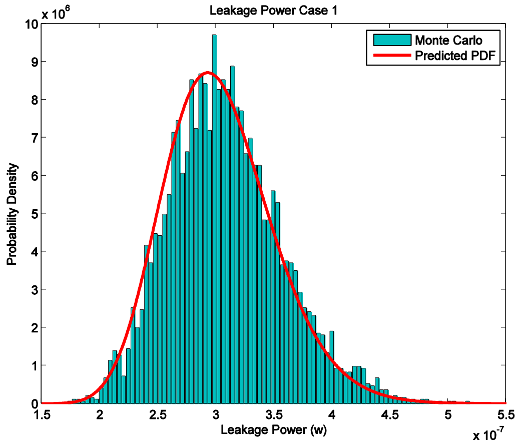


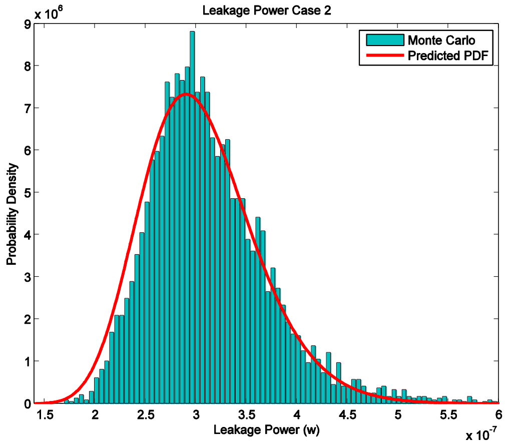


B.5 Delay Element

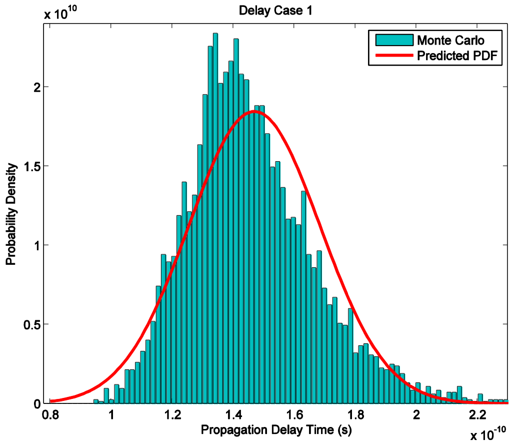


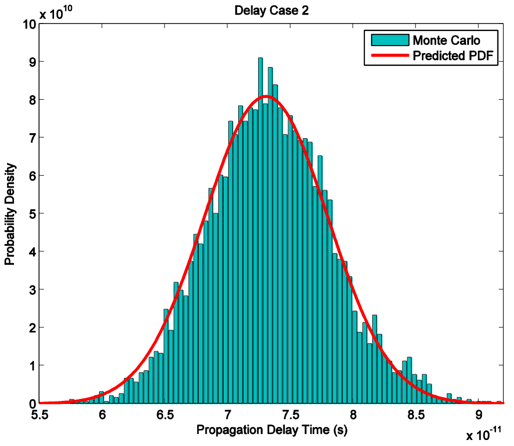


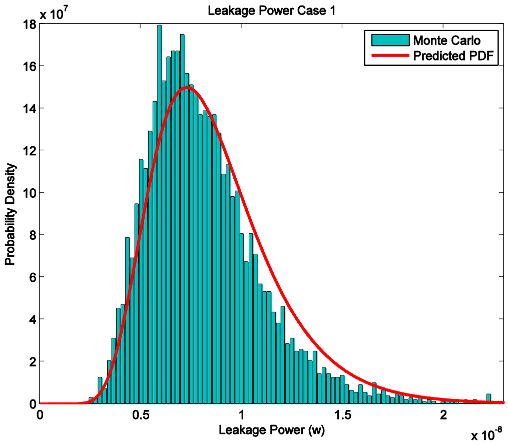


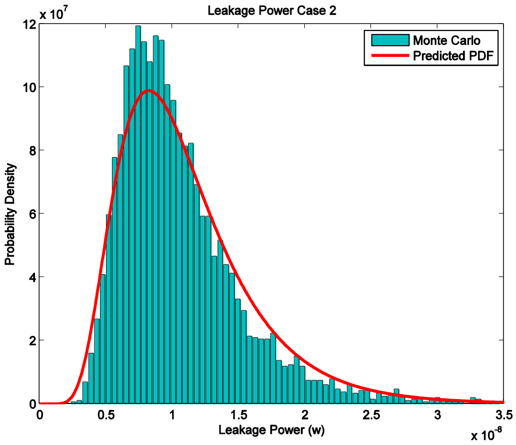


B.6 Muller-C Element






Leakage Power Case 1
x 10^7
18
16
14
12
10
8
6
4
2
0
Probability Density
Monte Carlo
Predicted PDF
0
0.5
1
1.5
2
Leakage Power (w)
x 10^-8


Leakage Power Case 2
x 10^7
12
10
8
6
4
2
0
Probability Density
Monte Carlo
Predicted PDF
0
0.5
1
1.5
2
2.5
3
3.5
Leakage Power (w)
x 10^-8