國立臺灣大學電機資訊學院電子工程學研究所

碩士論文

Graduate Institute of Electronics Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

考慮電源供應雜訊之動態時序分析器

Power-Supply-Noise-Aware Dynamic Timing Analyzer

謝弘毅

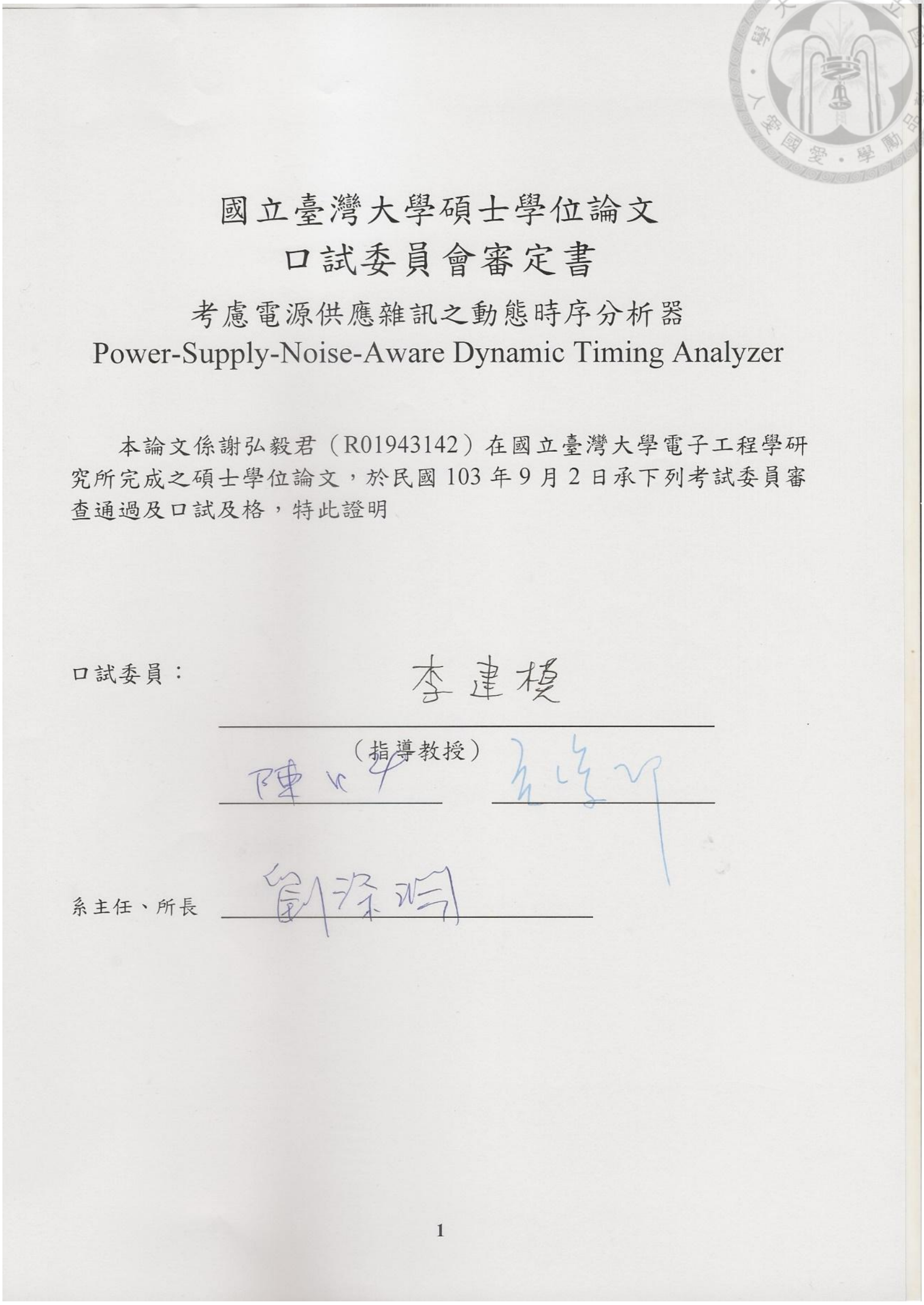
Hung-Yi Hsieh

指導教授：李建模 博士

Advisor: Dr. Li, Chien-Mo

中華民國 103 年 9 月

September, 2014



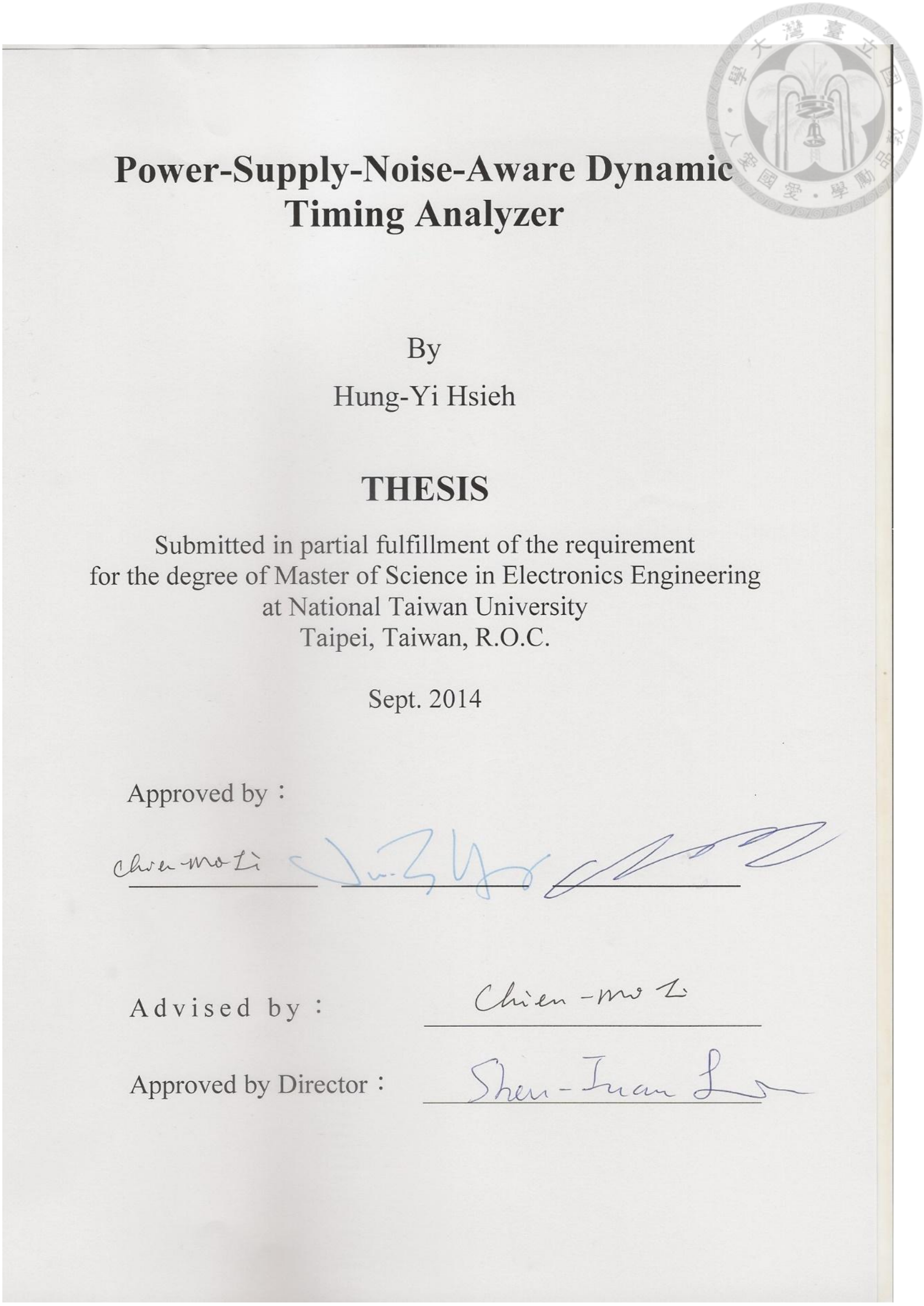
國立臺灣大學碩士學位論文
口試委員會審定書

考慮電源供應雜訊之動態時序分析器
Power-Supply-Noise-Aware Dynamic Timing Analyzer

本論文係謝弘毅君（R01943142）在國立臺灣大學電子工程學研究所完成之碩士學位論文，於民國 103 年 9 月 2 日承下列考試委員審查通過及口試及格，特此證明

口試委員：李建模
（指導教授）
陳

系主任、所長 劉深淵



Power-Supply-Noise-Aware Dynamic Timing Analyzer

By

Hung-Yi Hsieh

THESIS

Submitted in partial fulfillment of the requirement
for the degree of Master of Science in Electronics Engineering
at National Taiwan University
Taipei, Taiwan, R.O.C.

Sept. 2014

Approved by：

Chien-mo Li

Advised by： Chien-mo Li

Approved by Director： Shen-Iuan Liu

致謝

經過這兩年多的研究生活，首先要謝謝我的父母，有時候很晚回家，他們犧牲睡眠的等著我回家；有時候研究上不順利，他們也都承受著我的負面情緒，所以現在終於畢業了，最感謝的就是他們，讓我能無後顧之憂的努力學業。接著要謝謝我的指導教授，李建模教授，在研究上，老師扮演一個監督者，督促著進度並且讓我很耐心地跟我們討論；在職前訓練上，老師扮演一個很好的教練，不辭辛勞地一遍一遍糾正我的不足；在平時，老師十分關心大家的生活，體貼著大家的不方便。

接著要謝謝實驗室的博士班學長，首先是炳川學長，雖然跟學長只有幾面之緣，但碩二時，包含商業軟體使用的問題以及計畫報告書的撰寫上，幫忙很多也很大。再來是柏瑞學長，許多關於會議和計畫的問題還要是麻煩學長解答，而學長也總是不厭其煩得回答我們，此外，也要謝謝學長花時間替我解決許多投影片與口試上的問題。最後要謝謝官榆學長，許多程式與演算法上的問題，學長都能很細心的回答並給我們很好的意見。對於我們上一屆的學長，也非常的感謝。謝謝瑋陞學長，畢業前很熱心的跟我討論研究方向，畢業後還回來幫我修改口試頭影片。謝謝啟仁學長，在國科會計畫上留給我很多資源可以參考，在電子電路上也像活字典般有問必答。謝謝泓頤學長，在畢業前常推薦我很多很棒的電影，在畢業後也總是能在數學推導上跟我討論。謝謝介智學長，在我考研究所時給我很多參考資料，也不厭其煩的幫我解答許多疑問。謝謝聖章學長，在我當會計時幫忙很多，有任何問題都能告訴我該怎麼做或是可以提供我查詢的管道。

最要感謝的還是同一屆的各位同學。謝謝詩安，因為是同個大學的緣故，所以一開始有問題總是會找你幫忙，做研究時也常找你討論，修課時也真的多虧了你，我的研究生涯才能較為順利的過關。謝謝介甫，總是可以分擔我的怪，讓我不是實驗室最怪的人，也謝謝你常用美聲療癒大家的心靈。謝謝士閔，總是很細心的跟我們討論很多事情，並能提出很多不同的見解。謝謝昂鋒，總是寄信給大家提醒開會的時間。

最後要謝謝下一屆的學弟們，他們不僅要幫學長做一些事情，還要準備眾多的考試以及作業。其中最謝謝承佑學弟，在研究上幫助很多，幾乎可以說，沒有你就沒有這個成果，希望你之後研究順順利利，祝你早點畢業。至於其他學弟，謝謝大家的陪伴，帶給我許多歡笑。希望大家未來都能順順利利的完成學業，並進入理想的公司。

摘要

當測試超大型積體電路晶片時，由於電壓降和電感電壓的影響，電源供應雜訊會導致良率損失。在這篇論文中，我們提出一個考慮電源供應雜訊之動態時序分析器。我們提出的分析器提供合理的準確度和比現存工具還快的速度。因為我們提出的方法是基於線性函數而不是解非線性函數，所以是非常可調整的。實驗結果顯示：在小電路中，與 HSPICE 相比的誤差小於 90%，而速度快約 288,000 倍；在大電路中，我們達到比 NANOSIM 快八倍的速度，而誤差小於 50%。我們使用此分析器在一個有一百萬個邏輯閘的測試電路上，並且從三萬一千個測試向量中辨別出 12366 個時序違規的測試向量，這是傳統方法很難找得到的。

關鍵字：電源供應雜訊、電壓降、電感電壓、電荷、動態時序分析器。

Abstract

Due to the effect of IR-drop and Ldi/dt, power supply noise can cause yield loss when testing VLSI chips. In this thesis, we propose a power-supply-noise-aware dynamic timing analyzer, IDEA (IR-Drop-aware Efficient timing Analyzer). The proposed analyzer provides reasonable accuracy at much faster speed than existing tools. This technique is very scalable because it is based on linear functions, instead of solving nonlinear functions. The experimental results show, for small circuits, the error is less than 90% and the runtime is about 288,000 times shorter compared with HSPICE. For large circuits, we achieved eight times speed up compared with NANOSIM with error less than 50%. IDEA identifies 12366 timing-violation test patterns (out of 31K test patterns) for a 1M gate benchmark circuit which are difficult to detect by traditional techniques.

Key Words: power supply noise, IR-drop, Ldi/dt, charge, dynamic timing analyzer.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

1.1 Motivation

Power supply noise (PSN) becomes an important concern for VLSI system design and test [Shepard 1996][Saxena 2003][Wang 2005][Tehranipoor 2010]. PSN reduces the actual voltages supplied to logic gates, which also reduces signal integrity [Ma 2009]. Excessive PSN can degrade circuit performance by inducing extra gate delay, or even lead to timing failure of logic gates [Chen 1997][Jiang 1999]. It is also a well-known problem that excessive PSN during test can induce significant yield loss [Wang 2006][Li 2013]. Moreover, with technology scaling and power supply voltage lowering, path delay becomes more sensitive to power supply voltage, as shown in Figure 1.1 [Okumura 2010]. X axis is PSN (ΔV) and Y axis is *extra delay ratio*, which is the ratio of extra path delay to path delay ($\Delta D_{path}/D_{path}$). Figure 1.1 shows extra delay ratio at 45nm is about five times bigger than at 180nm when ΔV=0.2V.

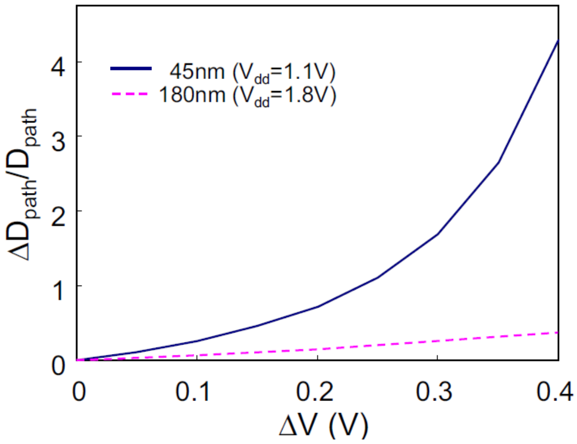
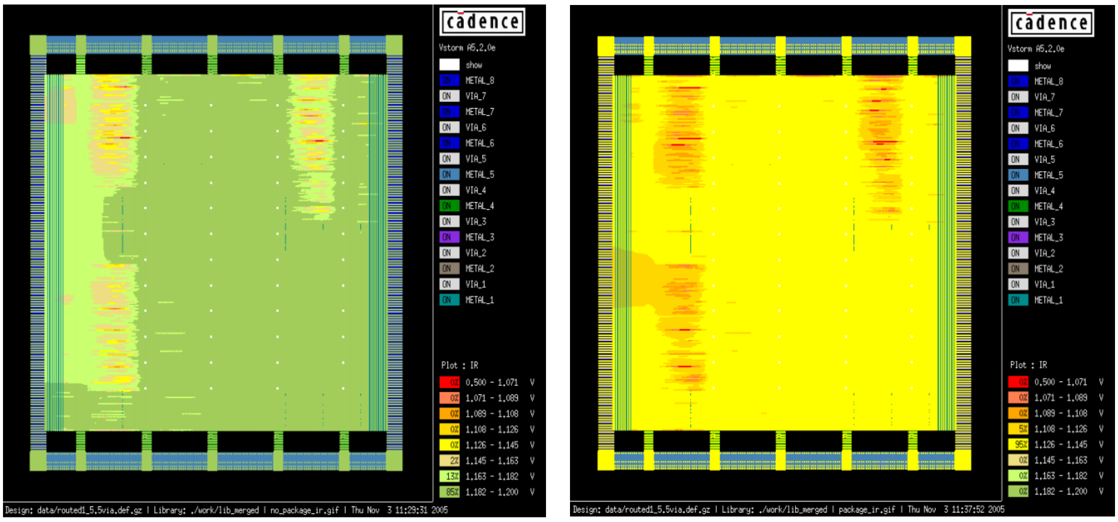


Figure 1.1 Comparison of extra delay ratio between 180nm and 45nm [Okumura 2010]

PSN can be classified into (1) *IR-drop* due to the parasitic resistances of on-chip interconnects, and (2) *Ldi/dt* due to package inductance. The first component (IR-drop) is a high-frequency noise, which is generated by switching gates. Traditional IR-drop analyzer shows the IR drop waveform or hot spot maps, but it is not clear how to translate IR-drop waveform to timing. The second component (*Ldi/dt*) is a mid-frequency noise, which is generated by off-chip inductance or package inductance [Ma 2011][Aparicio 2012]. Figure 1.2 compares IR-drop maps without package effects and with package effects. In Figure 1.2(a), the worst-case IR-drop without package effects is 147.5mV. In Figure 1.2(b), the worst-case IR-drop with package effects is 179.3mV. It can be seen that the effects of package need to be considered since we may overestimate circuit performance by ignoring package effects.



(a) (b)

Figure 1.2 IR-drop maps (a) without package effects and (b) with package effects

[Cadence 2009]

PSN-aware timing analysis can be classified into two classes. Static timing analysis does not require input patterns whereas dynamic timing analysis does. Static timing analysis is computationally efficient but it has problems to determine the value of PSN [Enami 2008]. Existing dynamic timing analysis tool is accurate but slow. For example, a commercial tool takes about twenty days to simulate all 31K test patterns for a million-gate benchmark circuit. Therefore, fast dynamic timing analysis for all test patterns is much needed to ensure both good test quality and low yield loss.

1.2 Proposed Technique

Figure 1.3 shows the concept of our approach to implement PSN-aware timing analysis. Since a single clock period is long, average PSN estimation for a whole clock period is not accurate enough. Therefore, we divide a clock period into non-overlapping equal-length windows. We sum up charges for every switching gate in this window, which divided by window width equals average current. We solve $\boldsymbol{G}\cdot\boldsymbol{V}+\boldsymbol{C}\cdot\boldsymbol{V}'=\boldsymbol{I}$ matrix to obtain average PSN in this window by KLU matrix solver [Davis 2010], where the $\boldsymbol{I}$ vector is obtained from average current. Finally, we use function of charges to translate average PSN to extra gate delay.

We model extra gate delay as a function of charges, which is stored in the output capacitor. Since the voltages supplied to logic gates determine the charges stored in the capacitor, the charges are the linear function of average PSN. Therefore, the impact of applying different current model is small.

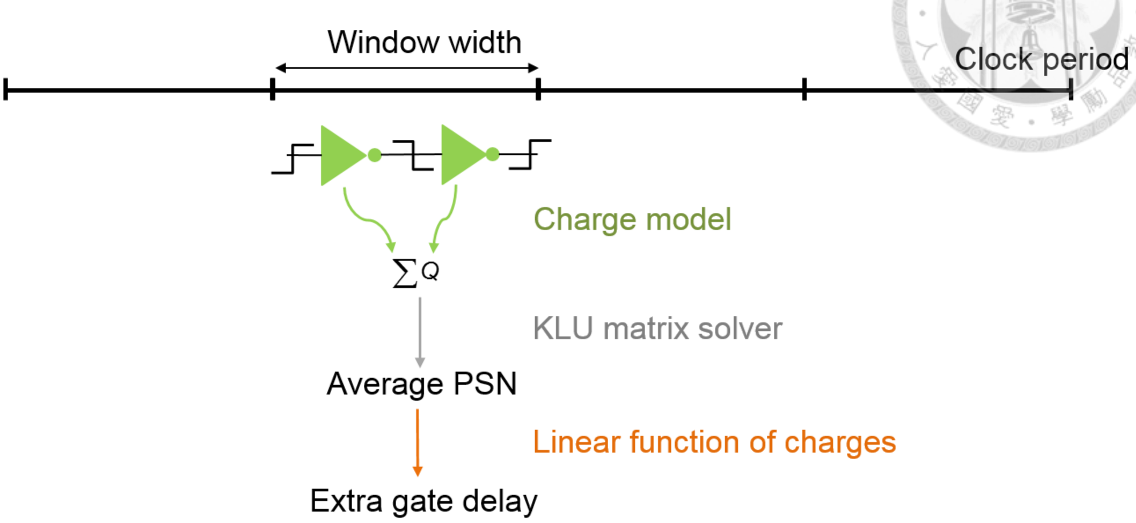


Figure 1.3 Concept of our approach

In this thesis, we propose a PSN-aware dynamic timing analyzer, *IDEA* (*IR-Drop-aware Efficient timing Analyzer*). Figure 1.4 shows the overall flow of IDEA. After performing *timed logic simulation*, the information of every switching gate is obtained. In *window partition*, IDEA partitions a clock period into non-overlapping windows. In every window, *charge model* for a switching gate is obtained from Synopsys library (*.lib*) file and is used in *average PSN estimation*. IDEA performs *extra gate delay estimation* for every switching gate in this window. If there is no more windows to process, IDEA produces PSN-aware path delay by *total path delay calculation*, which is the summation of all nominal gate delay and extra gate delay on the path. Finally, IDEA reports *path with maximum total path delay* for every test pattern.

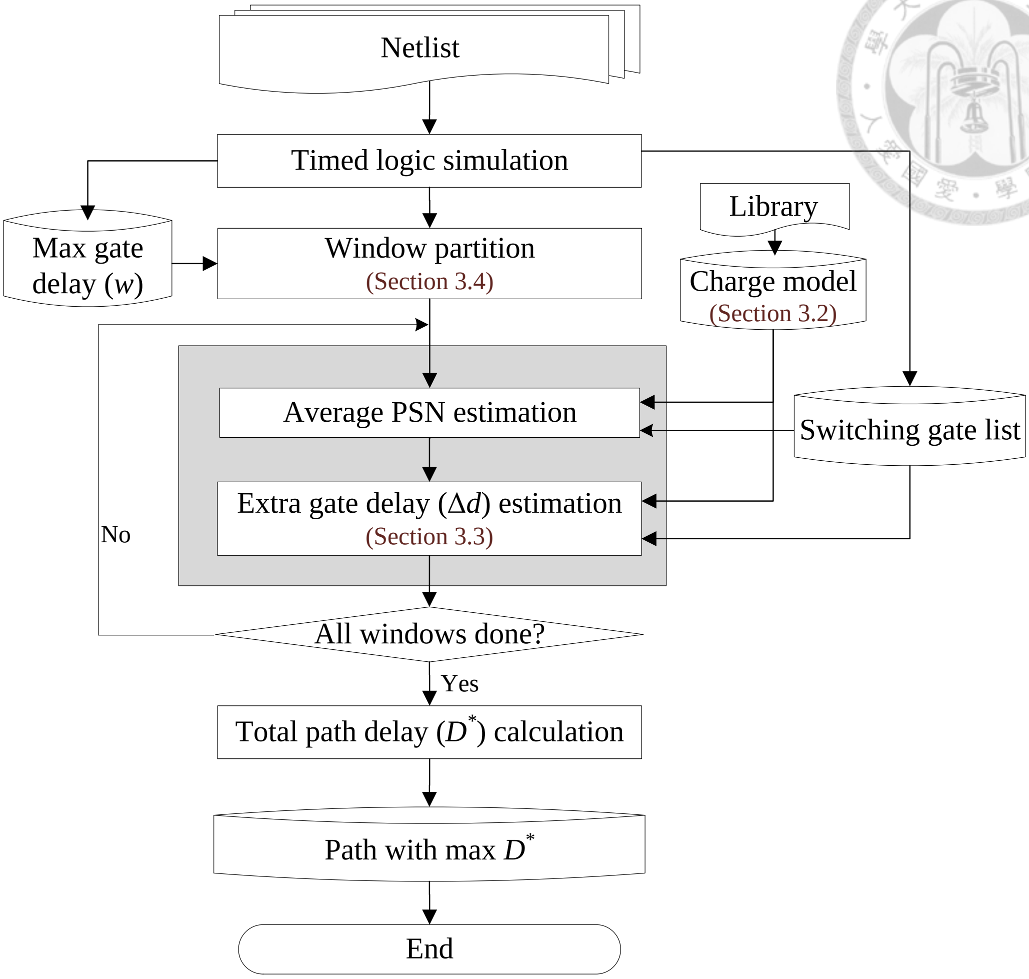


Figure 1.4 Overall flow of our proposed technique

In our experiments, IDEA has been applied to two cases. One case only considers the impact of IR-drop on path delay and the other considers both *Ldi/dt* and IR-drop. The results indicate the need for considering both *Ldi/dt* and IR-drop during dynamic timing analysis. Figure 1.5 shows path delay without PSN and with PSN (X axis) for the benchmark circuit *leon3mp* (1M gates). Y axis shows the number of test patterns in every interval. The histogram shows the importance of PSN since path delay increases significantly due to PSN.

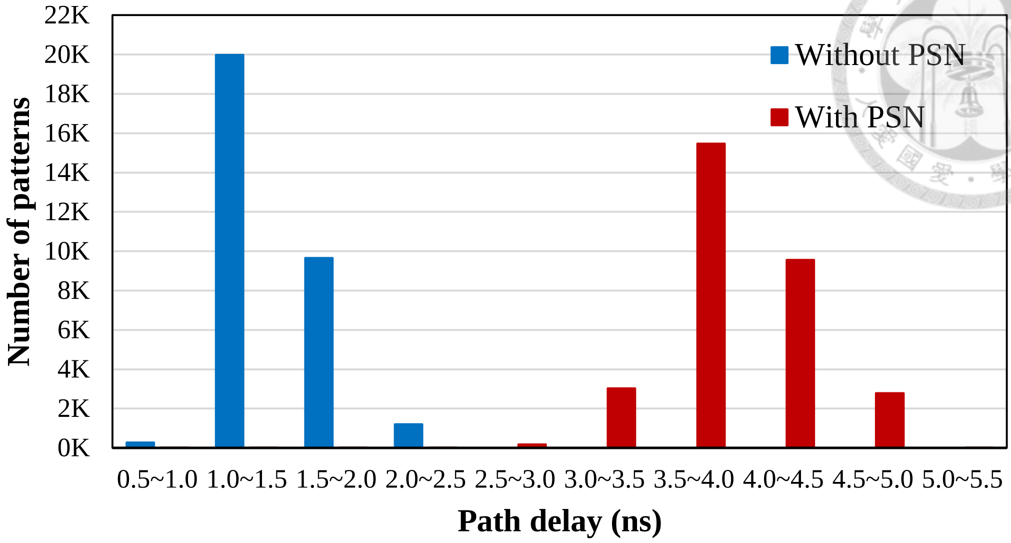


Figure 1.5 Histogram of path delay without PSN and with PSN (leon3mp)

Our tool has three advantages over traditional techniques. 1) IDEA models gate delay as linear equations, instead of nonlinear equations, so the runtime is very short. 2) IDEA models gate delay as a function of *charges*, instead of *voltage*, so that gate delay can be modeled accurately without database characterization. 3) IDEA considers both *Ldi/dt* and IR-drop altogether. In spite of the above, our tool has a limitation: the number of continuous clock cycles is limited by accumulated PSN error. The reason is that we use window width (about ninety times larger than a time step) as a time unit of simulation.

1.3 Contributions

This thesis has the following contributions to the research of PSN-aware dynamic timing analyzer.

- IDEA accurately estimates extra path delay, whose error is less than 1% compared with a commercial circuit simulator, HSPICE.
- IDEA achieves eight times speed up compared with a commercial tool, NANOSIM.
- IDEA models extra gate delay as functions of charge, so there is no characterization cost.
- IDEA dynamically analyzes PSN-induced extra delay by solving both *Ldi/dt* and IR-drop altogether.

1.4 Organization

The rest of the thesis is organized as follows. Chapter 2 reviews previous work about PSN estimation, extra gate delay calculation and PSN-aware timing analysis. Chapter 3 describes the details of IDEA. Chapter 4 shows experimental results on benchmark circuits. Chapter 5 is the discussion. Chapter 6 concludes this thesis.

Chapter 2 Background

It has been shown that PSN cannot be ignored during timing analysis [Liou 2003]. PSN-aware timing analysis consists of two steps: PSN estimation and extra gate delay calculation [Wang 2006]. Section 2.1 summarizes past researches about PSN estimation. Section 2.2 summarizes past researches about translation from PSN to extra gate delay. Section 2.3 summarizes past researches about PSN-aware timing analysis.

2.1 PSN Estimation

PSN is the noise on the power grid and ground grid, which is modeled as an RC or RLC network. However, for VLSI system design, circuit simulation of such a complicated network is infeasible, due to long runtime and memory limitation [Nassif 2000][Pant 2003][Wang 2006]. We summarize two solutions to estimate PSN without intensive computation. One solution is PSN model; the other is fast power grid analysis.

We review three simple PSN models, which are often used in past researches. In [Wen 2005][Wen 2007], *flip-flop toggle count* (FFTC) is defined as

$$FFTC = \sum_{i=1}^{N_F} S_i \tag{2.1}$$

, where S_i is the number of switches of flip-flop i and N_F is the total number of flip-flops. In [Ahmed 2007], *switching cycle average power* (SCAP), which is the average power consumed by a test pattern during the critical path delay (D), is defined a s

$$SCAP = \frac{\sum_{j=1}^{N_G} C_j \times VDD^2}{D} \tag{2.2}$$

, where C_j is the output capacitance of logic gate j, VDD is the nominal power supply voltage and N_G is the total number of logic gates. In [Girard 2002][Remersaro 2006], *weighted switching activity* (WSA) is defined as

$$WSA = \sum_{j=1}^{N_G} S_j F_j \tag{2.3}$$

, where S_j is the number of switches of logic gate j, F_j is the number of fan-out logic gates. These three metrics have no consideration on resistance and capacitance of the power grid, location of switching gates and power pads. Hence, it has been shown that these metrics do not correlate well with PSN [Varma 2012][Ding 2013].

The above simple PSN models are computationally efficient but inaccurate. Therefore, we introduce three power grid analyses for RC or RLC network with much shorter runtime than SPICE [Nassif 2000][Zhu 2003][Davis 2010].

In these three metrics, the analysis of RC or RLC network can be expressed as the following differential equation, which uses MNA formulation:

$$\boldsymbol{G} \cdot \boldsymbol{V} + \boldsymbol{C} \cdot \boldsymbol{V}' = \boldsymbol{I} \tag{2.4}$$

$\boldsymbol{G}$ is called *conductance matrix*. $\boldsymbol{C}$ includes the matrix of capacitance and inductance. $\boldsymbol{V}$ is the solution vector composed of node voltages and inductor currents. $\boldsymbol{V}'$ is the first derivative of $\boldsymbol{V}$ with respect to time. To obtain the solution, *backward Euler method*

(BE) is used to approximate $\boldsymbol{V}'$. Equations (2.5) to (2.7) show the derivation of applying BE method to equation (2.4). *h* is the time step size.

$$\boldsymbol{V}(t+h) = \boldsymbol{V}(t) + h \cdot \boldsymbol{V}'(t+h) \tag{2.5}$$

$$\boldsymbol{G} \cdot \boldsymbol{V}(t+h) + \boldsymbol{C} \cdot \frac{\boldsymbol{V}(t+h) - \boldsymbol{V}(t)}{h} = \boldsymbol{I}(t+h) \tag{2.6}$$

$$\boldsymbol{V}(t+h) = \left[\boldsymbol{G} + \frac{\boldsymbol{C}}{h}\right]^{-1} \cdot \left[\boldsymbol{I}(t+h) + \frac{\boldsymbol{C}}{h} \cdot \boldsymbol{V}(t)\right] \tag{2.7}$$

In equation (2.7), if *h* holds constant, only one initial matrix inversion is required. For large circuits, since the matrix inversion typically dominates the runtime of power grid analysis, the use of a constant *h* results in large savings.

A power grid reduction has been proposed in [Nassif 2000]. In all power grid nodes, only nodes at extremities of rows/columns and at intersection of a row and a column are kept, *kept nodes*; other nodes are removed, *removed nodes*. The nodes in reduced power grid are first solved. The voltage of a removed node is calculated by a linear function, which includes voltages of neighboring kept nodes and conductance between the removed node and neighboring kept nodes. Since the size of reduced power grid is much smaller than original power grid, runtime and memory needed are significantly reduced.

Due to the timing of switching gates, PSN exhibits spatial variation, which means some power grid nodes have more rapid voltage variations than other power grid nodes. An adaptive algebraic multigrid method has been proposed in [Zhu 2003]. The basic concept of multigrid is defining a hierarchy of a power grid, as shown in Figure 2.1. Every node at coarse grid level represents a set of nodes at fine grid level. In adaptive algebraic multigrid method, active regions with more PSN have relatively finer grid at coarse grid level since active regions need more computation to model their behavior accurately. The technique is used to speed up power grid analysis, taking advantage of the spatial variation of PSN.

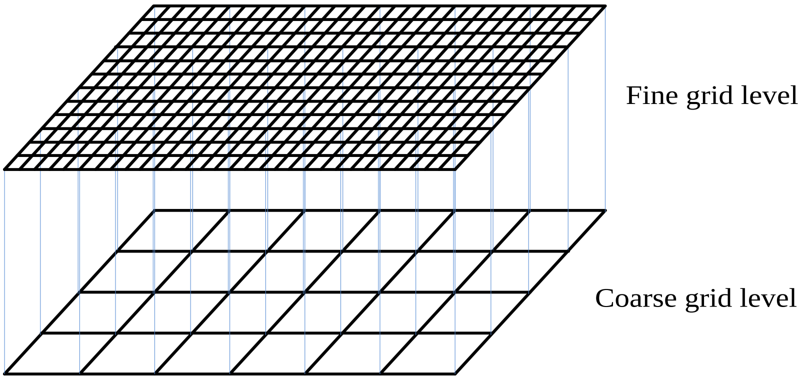


Figure 2.1 Concept of multigrid

These two techniques mentioned above are too expensive for practical use in estimating PSN [Wang 2006]. KLU is a sparse LU factorization algorithm, which can deal with sparse asymmetric matrices [Davis 2010]. KLU performs three steps. (1) The matrix is permuted into *Block Triangular Form* (BTF), a symmetric permutation that makes the matrix block upper triangular. (2) The *Approximate Minimum Degree* (AMD) ordering is used to fill-reducing order every block prior to LU factorization. (3) Gilbert/Peierls' left-looking LU factorization algorithm with partial pivoting is used to factorize every block. The total runtime is reduced since every block size is small.

2.2 Extra Gate Delay Calculation

PSN can induce extra gate delay and degrade circuit performance [Tehranipoor 2010]. We summarize four important techniques in recent research papers to calculate extra gate delay induced by PSN.

Extra gate delay is required to compute PSN, which is in turn required to compute extra gate delay. The first method proposed a procedure with iterative computation [Okumura 2010]. The procedure calculates average PSN during one time step at first, and then iteratively increases the number of time steps. Extra gate delay is calculated by a *voltage-delay characteristic function*, which is the function used to translate PSN to extra gate delay. After n iterations, if the difference between $n \times h$ and extra gate delay is smaller than h, the procedure finishes. h is the time step size. Since the time step is small, the method is accurate but slow.

For every gate in the library, SPICE simulation is performed under different conditions, such as transition type, power supply voltage of driver gate and receiver gate, input transition time, and output capacitance. The voltage-delay characteristic function can be stored in a database [Wang 2007][Aparicio 2013]. Translation from PSN to extra gate delay is done by table look-up, so runtime is short and the error is small. Since the database is obtained by intensive circuit simulation, the characterization cost is high.

The third method models the voltage-delay characteristic function as a regression polynomial function [Wang 2006][Todri 2012]. For every gate in the library, SPICE simulation is performed under different conditions, which is the same as the second method, in order to compute extra gate delay variations. Then coefficients for regression polynomials are calculated. Since intensive circuit simulation is needed, the advantage and the disadvantage is the same as the second method.

In the second method, SPICE simulation is performed under a lot of different conditions to build the database, so the characterization cost is extremely high. To avoid such intensive circuit simulation, the third method proposed the voltage-delay characteristic function using *equivalent output capacitor* [Hashimoto 2004] or *equivalent power supply voltage* [Hashimoto 2008], which is compatible with static timing analysis. Equivalent output capacitor and equivalent power supply voltage are used to reduce the number of parameters in the voltage-delay characteristic function. The goal of equivalent output capacitor is equalizing power supply voltages of driver gate and receiver gate, which causes charging/discharging current variation [Hashimoto 2004]. Equivalent output capacitor, which means increasing/decreasing the output capacitance in the same ratio as current variation, is used to keep the extra gate delay unchanged. Average power supply voltage is used as equivalent power supply voltage [Hashimoto 2008].

TABLE 2.1 shows the comparison of techniques among recent research papers.

The third column ‘Circuit’ shows circuits used in experimental results. The fourth column ‘Error’ shows the error compared with SPICE. These techniques were applied to small sample circuits only (such as NAND, INV and NOR).

TABLE 2.1 Comparison of previous translation from PSN to extra gate delay

Ref.	Method	Circuit	Error
[Hashimoto 2004]	Equivalent	Ten INV	Average error is 1.6%
[Hashimoto 2008]	Equivalent	Ten INV	Average error is 0.5%
[Okumura 2010]	Iterative	INV, NAND and NOR.	Error ranges from -2% to 2%
[Aparicio 2013]	Database	One INV	Maximal error is 0.35%
[Todri 2012]	Regression	Three INV	Error is 3.2%

2.3 PSN-Aware Timing Analysis

There are existing researches about PSN-aware timing analysis, which perform on circuits to ensure both good test quality and low yield loss. We summarize three important techniques in recent research papers for PSN-aware timing analysis.

Extra gate delay, which is obtained by the database, is used to update static standard delay format (*.sdf*) by considering PSN effect and generate pattern-dependent dynamic *.sdf* file for PSN-aware timing analysis [Peng 2010]. The database stores the voltage-delay characteristic function. Since the database is used to translate PSN to extra gate delay, characterization cost is high.

A gate-level event-driven simulator with two kinds of pre-characterized database is used for PSN-aware timing analysis [Jiang 2013]. *PM* is the first database, which is used to store PSN characteristic function, and *TM* is the second database, which is used to store the voltage-delay characteristic function. For every set of simultaneous events, PSN is calculated by PM and then extra gate delay is obtained by TM. The start time of the following events is updated by extra gate delay. There are two kinds of database that need to build, so characterization cost is much higher.

With performance and memory limitations of SPICE simulation, it is impossible for an entire VLSI system design. SPICE simulation is performed on critical paths, which is extracted by static timing analyzer, under transient PSN waveform [Apache 2011]. Both SPICE and transient PSN waveform simulation are accurate but slow when applied on big circuits. Besides, critical paths can change owing to extra gate delay induced by PSN, so the critical path delay obtained by this method may not be the worst case.

TABLE 2.2 shows the comparison of techniques among recent research papers. The second column ‘Technique’ shows the main concept of PSN-aware timing analysis. The third column ‘Method’ shows the main concept of translation from PSN to extra gate delay used in the technique. The fourth column ‘Circuit’ shows circuits used for timing analysis. The fifth column ‘Error’ shows PSN-aware path delay error compared with SPICE. Technique ‘Dynamic *.sdf* file’ showed only correlation but not accuracy [Peng 2010]. Technique ‘Two kinds of database’ used a benchmark circuit of 30K gates [Jiang 2013]. Their runtime for *p45* was 13 seconds per test pattern, which is still too slow for practical use. Therefore, there is still no general and efficient method to perform PSN-aware timing analysis so far.

TABLE 2.2 Comparison of previous PSN-aware timing analysis

Ref.	Technique	Method	Circuit	Error
[Peng 2010]	Dynamic *.sdf* file	Database	s344 (32 gates)	Correlation coefficient is 0.95
[Jiang 2013]	Two kinds of database	Database	p45 (30.6K)	N.A.
IDEA [This work]	Window partition	Linear function	b17 (32.5K)	Average error is 25.5%

Chapter 3 Proposed Techniques

We propose a new timing analyzer, IDEA, based on observations in Chapter 2. The advantages of IDEA are as follows. (1) We use windows, which is much larger than a time step but smaller than a clock period, to find good balance between accuracy and runtime. We do not need to calculate transient PSN for every time step. Instead, we only need to calculate average PSN in a window. Silicon data have been shown that average PSN correlates well with extra gate delay [Saint-Laurent 2004][Ogasahara 2007]. (2) IDEA models gate delay as a function of charges so that we do not need the voltage-delay characteristic function. There is no need for SPICE simulation and characterization. (3) IDEA is a dynamic timing analyzer, so the timing of every test pattern is considered accurately. In spite of the above, our tool has a limitation. The number of continuous clock cycles is limited by accumulated PSN error. As the number of clock cycles increases, the error of *Ldi/dt* increases. For more details, please see the Discussion Chapter.

3.1 Overall Flow

Figure 3.1 shows the overall flow of IDEA for a single test pattern.

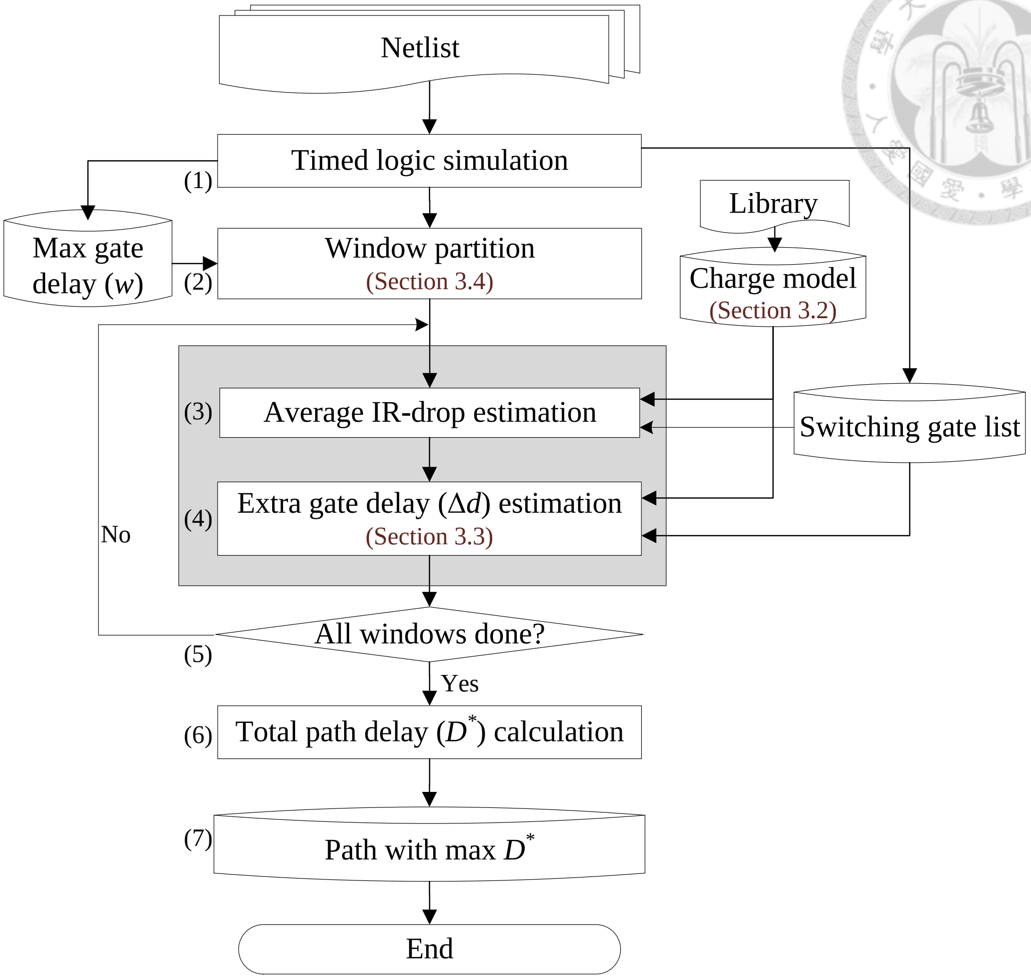


Figure 3.1 IDEA flow (for a single test pattern)

1) Perform timed logic simulation on the test pattern to obtain the information of every switching gate. Charge model for every switching gate, which will be detailed in Section 3.2, is obtained from the Synopsys library (.*lib*) file.

2) Use maximum gate delay as window width, *w*, which is used to set the boundary for every window. Window partition will be detailed in Section 3.4.

3) Select the first un-simulated window and then perform average PSN estimation for this window.

4) Perform extra gate delay (Δd) estimation to calculate PSN-induced extra gate delay for every switching gate in this window. Extra gate delay estimation will be detailed in Section 3.3.

5) If there is no more windows to process, move on to step 6; otherwise, continue the next un-simulated window and then repeat steps 3 and 4.

6) Calculate *total path delay* (D^*), which is obtained by

$$D^* = D + \Delta D \tag{3.1}$$

, where D is the path delay without PSN and ΔD is the PSN-induced extra path delay. ΔD is calculated by summing up Δd for every path.

7) Report path with maximum D^* for the test pattern. If D^* is larger than the clock period, this circuit may fail this test pattern owing to excessive PSN.

3.2 Charge Model

We use charge model to describe the relationship between PSN and extra gate delay. Charge model for every switching gate is used to calculate average current, average PSN and Δd for every window. The total energy consumed by a switching gate can be divided into *internal energy* and *switching energy* [Synopsys 2008]. Therefore, we separate Q into *internal charge* (Q_{IN}) and *switching charge* (Q_{SW}). The internal energy, which is consumed by short circuit current, is equal to $p\times\tau_I$. p is the internal power and τ_I is the input transition time of the switching gate, which can be looked up in the Synopsys library

(*.lib*) file. We use equation (3.2) to calculate Q_{IN}, where VDD is the nominal power supply voltage. The switching energy is consumed by charging or discharging the capacitor. We use equation (3.3) to calculate maximum charge stored in the capacitor (Q_{SW}), where C is the capacitance.

$$Q_{IN} = \frac{p \times \tau_I}{VDD} \tag{3.2}$$

$$Q_{SW} = C \times VDD \tag{3.3}$$

TABLE 3.1 shows average current in different conditions. Let P and G denote a power node and a ground node, respectively. Let R and F denote the output rising condition and the output falling condition, respectively. Every gate connects to P and G, as shown in Figure 3.2. $P0$, $P1$ and $P2$ are power nodes. $G0$, $G1$ and $G2$ are ground nodes.

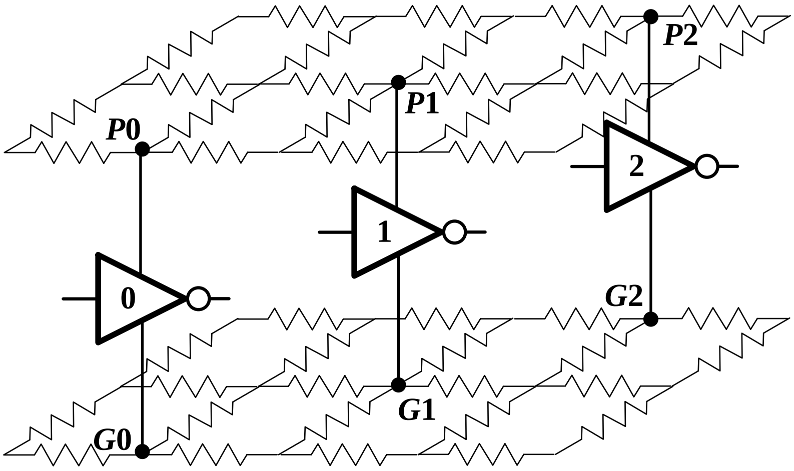
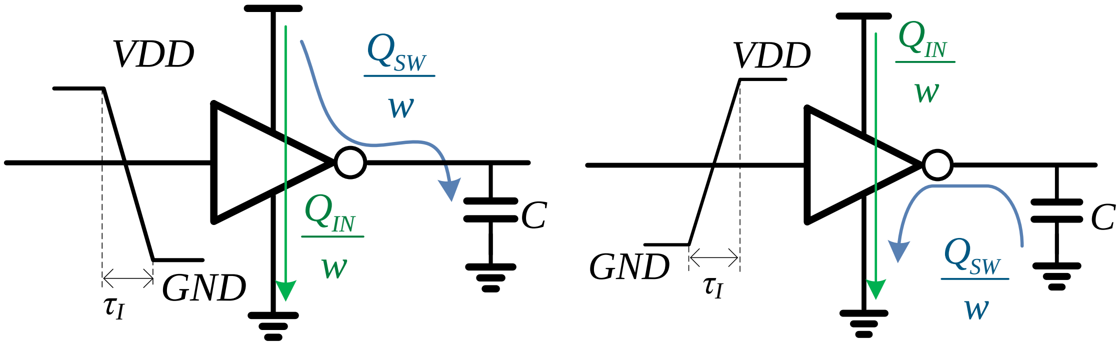


Figure 3.2 Example of power nodes and ground nodes

$\overline{I}_{PR}$ and $\overline{I}_{GR}$ are average rising current flowing out of P and average rising current flowing into G, respectively. $\overline{I}_{PF}$ and $\overline{I}_{GF}$ are average falling current flowing out of P and average falling current flowing into G, respectively. Figure 3.3 shows average current for output rising condition and output falling condition. In Figure 3.3(a), switching current for the output rising condition is flowing out of P to the capacitor, so Q_{SW}/w is only added to $\overline{I}_{PR}$, not to $\overline{I}_{GR}$. Similarly, in Figure 3.3(b), switching current for the output falling condition is flowing into G from the capacitor, so Q_{SW}/w is only added to $\overline{I}_{GF}$, not to $\overline{I}_{PF}$.



(a) (b)

Figure 3.3 Average current for (a) output rising and (b) output falling

TABLE 3.1 Average current cases

	Output rising	Output falling
Power node (current flows out of P)	$\overline{I}_{PR}=\frac{Q_{IN}+Q_{SW}}{w}$	$\overline{I}_{PF}=\frac{Q_{IN}}{w}$
Ground node (current flows into G)	$\overline{I}_{GR}=\frac{Q_{IN}}{w}$	$\overline{I}_{GF}=\frac{Q_{IN}+Q_{SW}}{w}$

3.3 Extra Gate Delay (Δ*d*) Estimation

Gate delay is the time between gate input transition and gate output transition, when they reach 50% VDD. Nominal gate delay (*d*) is the gate delay without PSN, which is obtained from the standard delay format (*.sdf*) file. Δ*d* is the PSN-induced extra gate delay. We need to estimate Δ*d* for every switching gate so that we can calculate the gate delay (d^*) under PSN effect.

$$d^* = d + \Delta d \tag{3.4}$$

Figure 3.4 shows an inverter with rising output. We use gate 1 and gate 2 to represent a *driver gate* and a *receiver gate*, respectively. In this thesis, we use this figure as an illustration example to estimate gate delay. v_{I1} and v_{I2} are input voltage of gate 1 and gate 2, respectively. v_{O1} and v_{O2} are output voltage of gate 1 and gate 2, respectively. They are functions of time, so they are denoted in small letters.

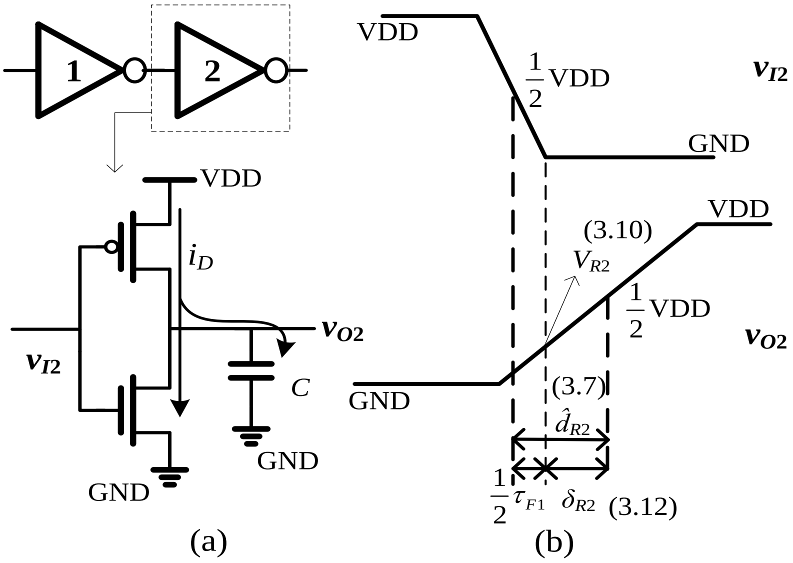


Figure 3.4 Example of rising gate delay estimation of gate 2

To estimate Δd_{R2}, which is the rising extra gate delay of gate 2, we use

$$\Delta d_{R2} = \overset{*}{d}_{R2} - d_{R2} \tag{3.5}$$

, where $\hat{d}_{R2}$ is the estimated rising gate delay of inverter 2 without PSN and $\hat{d}_{R2}^{*}$ is the estimated rising gate delay of inverter 2 with PSN. In this thesis, the hat symbol means the value is estimated and the asterisk symbol means the value is PSN-aware.

Figure 3.4(b) shows how to estimate $\hat{d}_{R2}$. Equation (3.6) is used in the estimation.

$$i_D = \frac{\beta}{2}(v_{GS} - V_{TH})^2 \tag{3.6}$$

, where i_D is the drain current through MOS, β is the *transconductance coefficient* of MOS, v_{GS} is the voltage between transistor gate and source and V_{TH} is the threshold voltage of MOS. Although we use level-1 quadratic model in this derivation, the conclusion of our work can be applied to other more accurate models. We will show that the conclusion is insensitive to the model in the Discussion Chapter. β and V_{TH} can be accessed in the MOS model, not in gate-level simulation. Therefore, two approximations are used to obtain Δd_{R2}, which will be detailed below.

i_D represents the current flowing out of *P*. One part of i_D is the short circuit current, which flows into *G*. The other part of i_D is switching current, which flows into the capacitor. The former is about a hundred times smaller than the latter. Therefore, we assume that switching current is equal to i_D.

In Figure 3.4, since v_{GS} changes during input transition, $\hat{d}_{R2}$ estimation is divided into two parts. One is the delay before v_{I2} reaches its GND; the other is the delay after v_{I2} reaches GND. The former is equal to half of input transition time of gate 2, which is equal to half of output transition time of gate 1. τ_{F1} is the falling output transition time of gate 1. The latter is defined as δ_{R2}.

$$d_{R2} = \frac{1}{2}\tau_{F1} + \delta_{R2} \tag{3.7}$$

τ_{F1} can be looked up from the *.lib* file, so we only need to calculate δ_{R2}. We use equations (3.8) to (3.10) to calculate $V_{R2,}$ which is the output voltage v_{O2}, when v_{I2} reaches GND. V_{R2} is a DC value, so it is denoted in capital letters.

$$C\frac{dv_{O2}}{dt} \approx i_D \tag{3.8}$$

$$C\int_{GND}^{V_{R2}} dv_{O2} \approx \frac{1}{2}\beta\int_0^{\tau_{F1}} (S_{I2}\times t - V_{TH})^2 dt \tag{3.9}$$

$$V_{R2} \approx \frac{\beta}{6\times C\times S_{I2}}(VDD - V_{TH})^3 \tag{3.10}$$

$$S_{I2} = \frac{VDD - GND}{\tau_{F1}} \tag{3.11}$$

S_{I2} is the input slope of gate 2, which can be derived from τ_{F1}.

Equation (3.12) calculates δ_{R2} which is based on $C\times\Delta v_{O2}/i_D = \Delta Q/i_D$. We can obtain the delay after v_{I2} reaches GND, as shown in equation (3.13), by substituting equation (3.10) into equation (3.12). δ_{R2} is measured as the delay from v_{O2}=V_{R2} to v_{O2}=0.5VDD.

$$\begin{aligned}\delta_{R2} &\approx \frac{C\times\Delta v_{O2}}{i_D} \\ &= \frac{C\left(\frac{1}{2}VDD - V_{R2}\right)}{\frac{1}{2}\beta\left(VDD - V_{TH}\right)^2}\end{aligned} \tag{3.12}$$

$$\delta_{R2} \approx \frac{\frac{1}{2}C \times VDD}{\frac{1}{2}\beta\left(VDD - V_{TH}\right)^{2}} - \frac{1}{3S_{I2}}\left(VDD - V_{TH}\right) \tag{3.13}$$

Second, we use similar way to calculate $\hat{d}^{*}_{R2}$, the estimated gate delay with PSN.

$$d^{*}_{R2} = \frac{1}{2}(\tau_{F1} + \Delta\tau_{F1}) + \delta^{*}_{R2} \tag{3.14}$$

In equation (3.14), δ^{*}_{R2} is the delay after v_{I2} reaches GND under PSN effect. S^{*}_{I2} is the input slope of gate 2 with PSN effect.

$$\delta^{*}_{R2} \approx \frac{C(\frac{1}{2}VDD - V_{L2})}{\frac{1}{2}\beta(V_{H2} - V_{L1} - V_{TH})^{2}} - \frac{(V_{H2} - V_{L1} - V_{TH})}{3S^{*}_{I2}} \tag{3.15}$$

$$S^{*}_{I2} = \frac{V_{H1} - V_{L1}}{\tau_{F1} + \Delta\tau_{F1}} \tag{3.16}$$

$\Delta\tau_{F1}$ is the PSN-induced extra falling output transition time of gate 1. The estimation of $\Delta\tau_{F1}$ will be detailed below.

In Figure 3.5, the waveform shows the output transition considering PSN. V_{H0} and V_{L0} are the power voltage of gate 0 and ground voltage of gate 0. V_{H1} and V_{L1} are the power voltage of gate 1 and ground voltage of gate 1. V_{H2} and V_{L2} are the power voltage of gate 2 and ground voltage of gate 2.

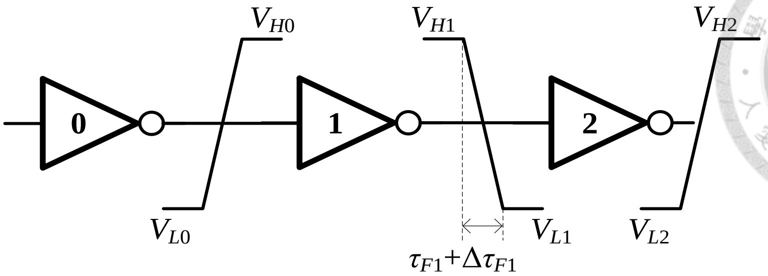


Figure 3.5 Example of I/O waveform considering PSN

To obtain the values of power voltage and ground voltage for every gate, we solve $\boldsymbol{G}\cdot\boldsymbol{V}+\boldsymbol{C}\cdot\boldsymbol{V}'=\boldsymbol{I}$ matrix to calculate average PSN and average ground bounce. The $\boldsymbol{I}$ vector is obtained from TABLE 3.1. Silicon data have been shown that average PSN correlates well with extra gate delay [Saint-Laurent 2004][Ogasahara 2007][Hashimoto 2008]. Values of V_{H0}, V_{H1} and V_{H2} can be substituted by VDD minus average PSN of the window. Values of V_{L0}, V_{L1} and V_{L2} can be substituted by average ground bounce of the window.

Finally, we calculate Δd_{R2} by substituting equation (3.7) and (3.14) into equation (3.5).

$$\Delta d_{R2}\approx\frac{C(\frac{1}{2}VDD-V_{L2})}{\frac{1}{2}\beta(V_{H2}-V_{L1}-V_{TH})^2}-\frac{\frac{1}{2}C\times VDD}{\frac{1}{2}\beta(VDD-V_{TH})^2}+ \\ \frac{(VDD-V_{TH})}{3S_{I2}}-\frac{(V_{H2}-V_{L1}-V_{TH})}{3S_{I2}^{*}}+\frac{\Delta\tau_{F1}}{2} \qquad (3.17)$$

V_{TH} cannot be accessed in gate-level simulation, so we need to remove it from equation (3.17). Since S_{I2} and S_{I2}^{*} are very large, we can make this approximation:

$$\frac{V_{TH}}{3S_{I2}^{*}}-\frac{V_{TH}}{3S_{I2}}\approx 0 \tag{3.18}$$

$$\Delta d_{R2}\approx \frac{C(\frac{1}{2}VDD-V_{L2})}{\frac{1}{2}\beta(V_{H2}-V_{L1}-V_{TH})^{2}}-\frac{\frac{1}{2}C\times VDD}{\frac{1}{2}\beta(VDD-V_{TH})^{2}}+\frac{VDD}{3S_{I2}}-\frac{(V_{H2}-V_{L1})}{3S_{I2}^{*}}+\frac{\Delta\tau_{F1}}{2} \tag{3.19}$$

Output transition time is the time between GND and VDD of gate output transition. Nominal output transition time (τ) is the output transition time without PSN, which can be obtained from the *.lib* file. $\Delta\tau$ is the PSN-induced extra output transition time, which is needed for estimating output transition time (τ^{*}) under PSN effect.

We use a model to calculate output transition time, which is proposed in [Maurine 2001]. In equation (3.20), $\hat{\tau_{F1}}$ and $\hat{\tau_{F1}^{*}}$ are the estimated falling output transition time of inverter 1 without PSN and with PSN, respectively.

$$\begin{aligned}\Delta\tau_{F1} &= \hat{\tau}^{*}_{F1}-\hat{\tau}_{F1}\\ &\approx \frac{C(V_{H1}-V_{L1})}{\frac{1}{2}\beta(V_{H0}-V_{L1}-V_{TH})^{2}}-\frac{C(VDD-GND)}{\frac{1}{2}\beta(VDD-V_{TH})^{2}}\end{aligned} \tag{3.20}$$

In equation (3.19) and (3.20), the values of β and V_{TH} are not determined yet. Thus we use *peak current* to replace the current in these equations, like equations (3.21) and (3.22).

$$\Delta d_{R2}\approx \frac{\Delta\tau_{F1}}{2}+\frac{C\left(\frac{1}{2}VDD-V_{L2}\right)}{\tilde{I}^{*}_{PR2}}-\frac{\frac{1}{2}C\times VDD}{\tilde{I}_{PR2}}+\frac{VDD}{3S_{I2}}-\frac{(V_{H2}-V_{L1})}{3S_{I2}^{*}} \tag{3.21}$$

$$\Delta\tau_{F1} \approx C\times\left(\frac{V_{H1}-V_{L1}}{\tilde{I}^{*}_{GF1}}-\frac{VDD-GND}{\tilde{I}_{GF1}}\right) \tag{3.22}$$

$\tilde{I}_{PR2}$ and $\tilde{I}^{*}_{PR2}$ are peak current for output rising of gate 2 without PSN and with PSN, respectively. $\tilde{I}_{GF1}$ and $\tilde{I}^{*}_{GF1}$ are peak current for output falling of gate 1 without PSN and with PSN, respectively.

To calculate the value of peak current, we use the equalization of charge to explain the derivation. Equation (3.23) shows the integral of i_D and we assume d_{R2}≫$0.5\tau_{F1}$. One part of i_D is the short circuit current. Since the duration of d_{R2} only include half of input transition time, the charge is equal to 0.5×Q_{IN}. The other part of i_D flows through the capacitor for charging. Since the range of v_{O2} variation during d_{R2} is 0.5×VDD, the charge is equal to 0.5×Q_{SW}. Therefore, Q_D is equal to 0.5×(Q_{IN}+Q_{SW}). $\bar{I}_{PR2}$ is $\bar{I}_{PR}$ of gate 2.

$$\begin{aligned}
Q_D &= \int_0^{d_{R2}}\left[\frac{\beta}{2}\left(v_{GS}-V_{TH}\right)^2\right]dt \\
&= \int_0^{\frac{1}{2}\tau_{F1}}\left[\frac{\beta}{2}\left(v_{GS}-V_{TH}\right)^2\right]dt+\int_{\frac{1}{2}\tau_{F1}}^{d_{R2}}\left[\frac{\beta}{2}\left(VDD-V_{TH}\right)^2\right]dt \\
&\approx d_{R2}\times\frac{\beta}{2}\left(VDD-V_{TH}\right)^2 = Q_D
\end{aligned} \tag{3.23}$$

$$\frac{Q_D}{w}\approx\frac{Q_D}{w}=\frac{Q_{SW}+Q_{IN}}{2w}=\frac{1}{2}\bar{I}_{PR2} \tag{3.24}$$

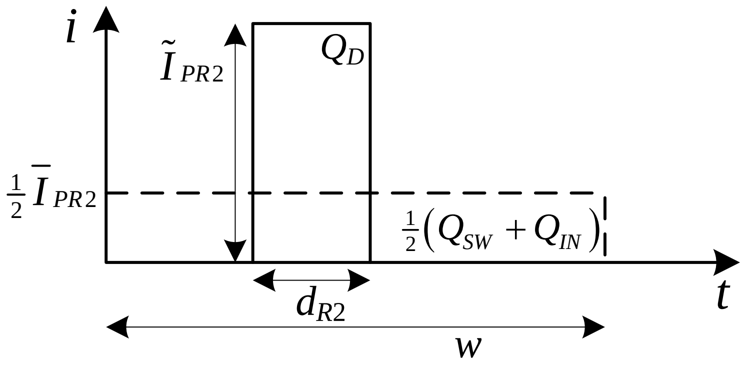
We substitute equation (3.23) into equation (3.24) and obtain

$$\frac{d_{R2}}{w}\times\frac{\beta}{2}\left(VDD-V_{TH}\right)^2\approx\frac{1}{2}\bar{I}_{PR2} \tag{3.25}$$

Therefore, $\tilde{I}_{PR2}$ in equation (3.21) is obtained.

$$\tilde{I}_{PR2} = \bar{I}_{PR2} \times \frac{w}{2d_{R2}} \tag{3.26}$$

Figure 3.6 shows the current waveform of $\bar{I}_{PR2}$ and $\tilde{I}_{PR2}$. The area of these two rectangles presents charges. Q_{SW} and Q_{IN} can be calculated from the *.lib* file. Since Q_D is equal to 0.5×(Q_{IN}+Q_{SW}), the two rectangles are the same in area. They are different by the width. One is window width *w*, the other is gate delay d_{R2}.



i
$\tilde{I}_{PR2}$
Q_D
$\frac{1}{2}\bar{I}_{PR2}$
$\frac{1}{2}\left(Q_{SW}+Q_{IN}\right)$
t
d_{R2}
w

Figure 3.6 Current waveform transformation

Similarly, $\tilde{I}^*_{PR2}$, $\tilde{I}_{GF1}$ and $\tilde{I}^*_{GF1}$ in equations (3.21) and (3.22) are calculated by:

$$\tilde{I}^*_{PR2} = \frac{p\times\tau_{F1}}{2\left(V_{H1}-V_{L1}\right)d_{R2}} + \frac{C\left(V_{H2}-V_{L2}\right)}{2d_{R2}} \tag{3.27}$$

$$\tilde{I}_{GF1} = \frac{p\times\tau_{R0}}{2\times VDD\times d_{F1}} + \frac{C\times VDD}{2d_{F1}} \tag{3.28}$$

$$\tilde{I}^*_{GF1} = \frac{p\times\tau_{R0}}{2\left(V_{H0}-V_{L0}\right)d_{F1}} + \frac{C\left(V_{H1}-V_{L1}\right)}{2d_{F1}} \tag{3.29}$$

We substitute equations (3.37) to (3.39) into equations (3.21) and (3.22) and obtain equations (3.30) and (3.31).

$$\Delta d_{R2} \approx \frac{\Delta\tau_{F1}}{2} + \frac{C\left(\frac{1}{2}VDD - V_{L2}\right)}{\frac{p \times \tau_{F1}}{2\left(V_{H1} - V_{L1}\right)d_{R2}} + \frac{C\left(V_{H2} - V_{L2}\right)}{2d_{R2}}} - \frac{\frac{1}{2}C \times VDD}{\frac{p \times \tau_{F1}}{2 \times VDD \times d_{R2}} + \frac{C \times VDD}{2d_{R2}}} + \frac{VDD}{3S_{I2}} - \frac{(V_{H2} - V_{L1})}{3S_{I2}^{*}} \tag{3.30}$$

$$\Delta\tau_{F1} \approx C \times \left(\frac{V_{H1} - V_{L1}}{\frac{p \times \tau_{R0}}{2\left(V_{H0} - V_{L0}\right)d_{F1}} + \frac{C\left(V_{H1} - V_{L1}\right)}{2d_{F1}}} - \frac{VDD - GND}{\frac{p \times \tau_{R0}}{2 \times VDD \times d_{F1}} + \frac{C \times VDD}{2d_{F1}}} \right) \tag{3.31}$$

In these two equations, we model extra gate delay and extra output transition time as function of charges, but not current model. Therefore, the impact of applying different drain current model is small.

We use similar way to estimate Δd_{F2} and $\Delta\tau_{R1}$, as shown in equations (3.32) and (3.33).

$$\Delta d_{F2} \approx \frac{\Delta\tau_{R1}}{2} + \frac{C\left(V_{H2} - \frac{1}{2}VDD\right)}{\tilde{I}_{GF2}^{*}} - \frac{\frac{1}{2}C \times VDD}{\tilde{I}_{GF2}} + \frac{VDD}{3S_{I2}} - \frac{(V_{H1} - V_{L2})}{3S_{I2}^{*}} \tag{3.32}$$

$$\Delta\tau_{R1} \approx C \times \left(\frac{V_{H1} - V_{L1}}{\tilde{I}_{PR1}^{*}} - \frac{VDD - GND}{\tilde{I}_{PR1}} \right) \tag{3.33}$$

,where $\tilde{I}_{GF2}$ and $\tilde{I}_{GF2}^{*}$ are peak current for output falling of gate 2 without PSN and with PSN, respectively. $\tilde{I}_{PR1}$ and $\tilde{I}_{PR1}^{*}$ are peak current for output rising of gate 1 without PSN and with PSN, respectively.

$$\tilde{I}_{GF2} = \frac{p \times \tau_{R1}}{2 \times VDD \times d_{F2}} + \frac{C \times VDD}{2d_{F2}} \tag{3.34}$$

$$\tilde{I}_{GF2}^{*} = \frac{p \times \tau_{R1}}{2\left(V_{H1} - V_{L1}\right)d_{F2}} + \frac{C\left(V_{H2} - V_{L2}\right)}{2d_{F2}} \tag{3.35}$$

$$\tilde{I}_{PR1} = \frac{p \times \tau_{F0}}{2 \times VDD \times d_{R1}} + \frac{C \times VDD}{2d_{R1}} \tag{3.36}$$

$$\tilde{I}^{*}_{PR1} = \frac{p \times \tau_{F0}}{2\left(V_{H0} - V_{L0}\right) d_{R1}} + \frac{C\left(V_{H1} - V_{L1}\right)}{2d_{R1}} \tag{3.37}$$

3.4 Window Partition

Since a single clock period is long, average PSN estimation for a whole clock period is not accurate enough. According to [Devanathan 2007][Wen 2008][Wu 2010], the window partition improves the average PSN estimation quality because the temporal requirement of switching gates is taken into consideration. Therefore, we divide a whole clock period into several non-overlapping equal-length time slices, called *windows*.

We need to decide the window width, *w*. If *w* is too large, average PSN is very low so Δ*d* can be underestimated. On the contrary, if *w* is too small, we see a scenario where *d* of a switching gate crosses window boundaries. Figure 3.7 illustrates such a scenario. d_1 is the partial gate delay in window 1 and d_2 is the partial gate delay in window 2. For such a scenario, it is not clear that the charge of this switching gate contributes to which window.

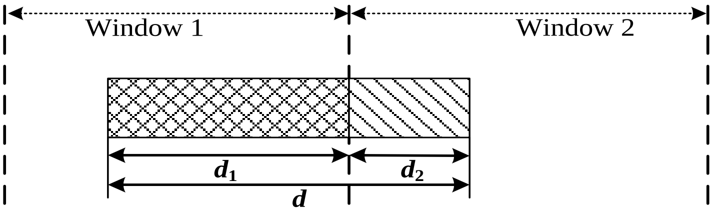


Figure 3.7 Switching gate delay crosses a window boundary

In this thesis, we propose to use maximum gate delay as window width *w*. In this way, we ensure that *d* of every switching gate crosses at most one window boundary. For the switching gate that crosses a window boundary, we use a weighted ratio to compute the contribution of its charges. In equations (3.38) and (3.39), Q_1 contributes to window 1 and Q_2 contributes to window 2. Q is equal to Q_{IN}+Q_{SW} or Q_{IN}, which is determined by four cases in TABLE 3.1. We use Q_1 and Q_2 in average PSN estimation.

$$Q_1 = Q \times \frac{d_1}{d} \tag{3.38}$$

$$Q_2 = Q \times \frac{d_2}{d} \tag{3.39}$$

In this thesis, we propose to use dynamic window partition, where each pattern has its own number of windows. We will show that dynamic window partition is better than static window partition, where all patterns use the same number of windows.

Chapter 4 Experimental Results

4.1 Experimental Setup

To demonstrate the accuracy and effectiveness of our proposed technique, experiments are performed on ISCAS'89, ITC'99 and IWLS'05 benchmark circuits, which are mapped to NanGate 45nm technology (nominal VDD=1.1V). The circuits are placed and routed by Cadence *SOC Encounter*.

TABLE 4.1 shows the basic information of benchmark circuits. The largest circuit, *leon3mp*, has two pairs of VDD/GND power pad while the other circuits only have one pair of power pad. Vertical and horizontal power stripes are added to *leon3mp*. Figure 4.1 shows the VDD/GND power grid of each benchmark circuit.

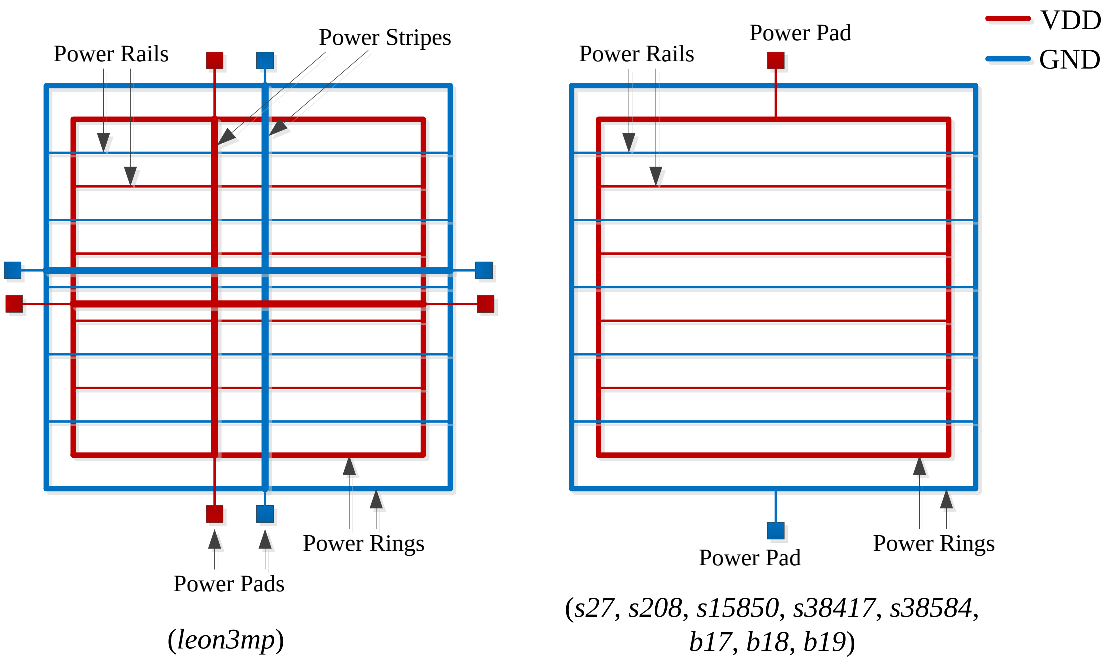


Figure 4.1 VDD/GND power grid

The fourth column '# Extracted nodes' is obtained from the power grid RC models, which are extracted by Cadence *QRC*. Launch-on-capture transition fault test patterns are generated by Synopsys *TetraMAX ATPG*. The sixth column 'Clock period' shows the clock period with 15% margins. These experiments are conducted on a Linux system, which has 3.4GHz CPU with 32GB memory.

TABLE 4.1 Benchmark circuits

Circuit	# Gates	# FFs	# Extracted nodes	Test length	Clock period (ns)
s27	16	3	28	10	1
s208	70	8	128	43	2
s15850	2.9K	510	5.8K	151	3
s38417	8.5K	1.6K	16.5K	185	4
s38584	8.7K	1.3K	14.2K	319	4
b17	32.5K	1.4K	49.4K	1.4K	4
b18	73.0K	3.3K	92.0K	2.0K	4
b19	147.1K	6.5K	155.8K	2.4K	4
leon3mp	1.0M	108.8K	744.5K	31.1K	4

4.2 IR-drop Only Experiments

TABLE 4.2 compares the experimental results of extra path delay in IDEA, ΔD, and in HSPICE simulation, ΔD_{TOOL}. We run IDEA to obtain a critical path with maximum total path delay (D^{*}) of every test pattern for every benchmark circuit. *KLU matrix solver* [Davis 2010] is used to solve $\boldsymbol{G} \cdot \boldsymbol{V} + \boldsymbol{C} \cdot \boldsymbol{V}' = \boldsymbol{I}$ matrix for every window. ΔD is obtained from equation (3.1). We perform HSPICE simulation to extract the critical path delay in two cases. One case is the design with nominal power supply voltage and the other is with power grid RC model (D^{*}_{TOOL}). ΔD_{TOOL} is the difference between the values of two path delay. The second column ‘HSPICE’ shows the average runtime of HSPICE simulation with power grid RC model. The third column ‘Setup’ shows the runtime to build the *conductance matrix* and to inverse the matrix by KLU. We only need to perform the setup once for every circuit. The fourth column ‘Simulation’ shows the average runtime per test pattern. The fifth and sixth columns ‘E^{Δ}’ show the extra path delay error of IDEA with respect to HSPICE, calculated by equation (4.1). The seventh and eighth columns ‘E^{*}’ show the total path delay error of IDEA with respect to HSPICE, calculated by equation (4.2). The table shows that average E^{Δ} is less than 125% and the runtime of IDEA is about 288,000 times faster than HSPICE.

$$E^{\Delta}(\%) = \frac{\Delta D - \Delta D_{TOOL}}{\Delta D_{TOOL}} \times 100(\%) \tag{4.1}$$

$$E^{*}(\%) = \frac{D^{*} - D^{*}_{TOOL}}{D^{*}_{TOOL}} \times 100(\%) \tag{4.2}$$

TABLE 4.2 Experimental results of path delay

Circuit	Runtime			E^{Δ} (%)		E^{*} (%)	
	HSPICE (s/pat)	IDEA		Max.	Avg.	Max.	Avg.
		Setup (s)	Simulation (s/pat)				
s27	0.9	0	0	102.3	55.0	16.7	3.3
s208	9.9	0	0	123.1	80.6	29.6	6.1
s15850	2,389	0	0	112.3	59.3	87.5	7.6
s38417	13,352	0	0.1	107.8	74.0	43.1	26.0
s38584	54,443	0	0.1	110.6	78.4	72.0	29.3
b17	81,772	0	0.1	108.4	73.5	68.7	25.5

HSPICE cannot run big circuits so we use NANOSIM instead. TABLE 4.3 compares the experimental results of total path delay in NANOSIM simulation, D^{*}_{TOOL}, and IDEA, D^{*}. D^{*}_{TOOL} is the critical path delay, extracted by IDEA, reported by NANOSIM.

The second column 'Setup' shows the runtime of library compilation and circuit partition, which is only performed once for every circuit. The third column 'Simulation' shows the average runtime of NANOSIM with power grid RC model. The sixth and seventh columns 'E^{*}' show the total path delay error of IDEA with respect to NANOSIM, calculated by equation (4.2). TABLE 4.3 shows that average E^{*} is less than 45% and runtime is 8.4 times faster with respect to NANOSIM. In *leon3mp*, the total runtime of IDEA for all test patterns is about four days while the total runtime of NANOSIM is about twenty days. A positive error means that our path delay estimation is larger than

NANOSIM. Please note that E^{Δ} and E^{*} are always positive so the results are pessimistic.

TABLE 4.3 Experimental results of total path delay

Circuit	Runtime				E^{*} (%)	
	NANOSIM		IDEA		Max.	Avg.
	Setup (s)	Simulation (s/pat)	Setup (s)	Simulation (s/pat)		
b18	46.0	4.9	0	0.5	68.8	31.7
b19	95.2	9.6	0.1	1.0	64.2	26.8
leon3mp	734.77	61.05	0.46	11.40	84.3	44.2

The sources of errors can be summarized as follows:

1) Since drain current i_D flows through the capacitor, i_D determine the speed of charging/discharging, which also determine gate delay. However, in equations (4.3) and (4.4), we use peak current to estimate extra gate delay.

$$\Delta d_{R2} \approx \frac{\Delta\tau_{F1}}{2} + \frac{C\left(\frac{1}{2}VDD - V_{L2}\right)}{\tilde{I}^{*}_{PR2}} - \frac{\frac{1}{2}C \times VDD}{\tilde{I}_{PR2}} + \frac{VDD}{3S_{I2}} - \frac{(V_{H2} - V_{L1})}{3S^{*}_{I2}} \tag{4.3}$$

$$\Delta d_{F2} \approx \frac{\Delta\tau_{R1}}{2} + \frac{C\left(V_{H2} - \frac{1}{2}VDD\right)}{\tilde{I}^{*}_{GF2}} - \frac{\frac{1}{2}C \times VDD}{\tilde{I}_{GF2}} + \frac{VDD}{3S_{I2}} - \frac{(V_{H1} - V_{L2})}{3S^{*}_{I2}} \tag{4.4}$$

i_D is proportional to $(v_{GS}$-$V_{TH})^2$, as shown in equation (4.5), but peak current is proportional to IR-drop.

$$i_D = \frac{\beta}{2}(v_{GS} - V_{TH})^2 \tag{4.5}$$

We use Figure 4.2 to demonstrate the difference between i_D and peak current. We simulate the falling transition of an INV_X1 gate. During the experiment, i_D of

NMOS is measured comparing to peak current, which is obtained by equation (3.35). The figure shows that as IR-drop increases, the difference becomes larger.

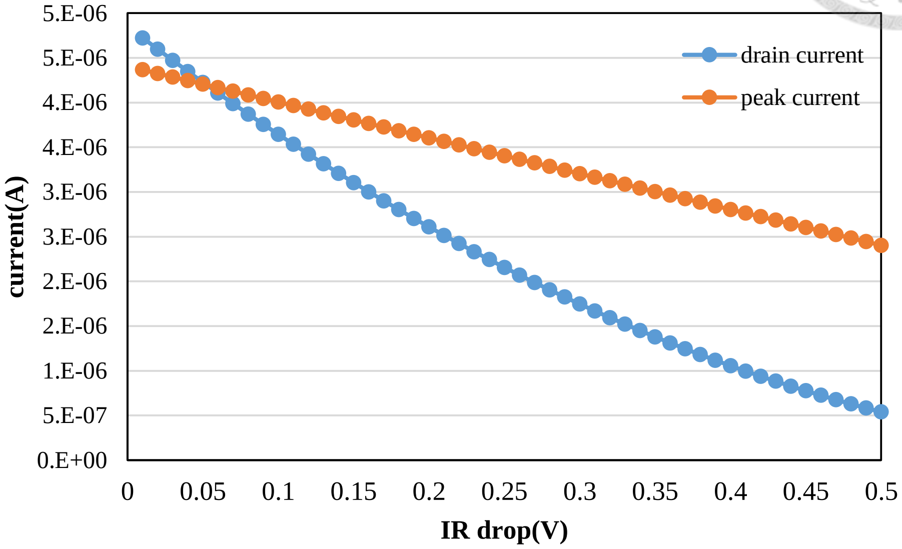


Figure 4.2 Difference between drain current and peak current

Since we use peak current to replace i_D, the effect of v_{GS} cannot be considered properly, which leads to errors in delay estimation.

2) In the estimation of extra output transition time, we also use peak current to replace i_D. We take equation (4.6) as an example to illustrate the effect of replacement. In equation (4.7), which is obtained by simplifying equation (4.6), extra output transition time is always negative. Like 1), since the replacement, the effect of v_{GS} cannot be considered properly, which leads to errors in delay estimation.

$$\Delta\tau_{F1} \approx C \times \left(\frac{V_{H1} - V_{L1}}{\frac{p \times \tau_{R0}}{2\left(V_{H0} - V_{L0}\right) d_{F1}} + \frac{C\left(V_{H1} - V_{L1}\right)}{2d_{F1}}} - \frac{VDD - GND}{\frac{p \times \tau_{R0}}{2 \times VDD \times d_{F1}} + \frac{C \times VDD}{2d_{F1}}} \right) \quad (4.6)$$

$$\Delta\tau_{F1} \approx 2d_{F1} \times C \times \left(\frac{\left(V_{H0} - V_{L0}\right)\left(V_{H1} - V_{L1}\right) - VDD^2}{p \times \tau_{R0}} \right) \tag{4.7}$$

3) In average IR-drop estimation, we use average current in a window to replace transient current for every time step. Since the window width we use is about ninety times larger than a time step, the negligence of time-variant current may lead to errors in delay estimation.

4) In our extra gate delay estimation, we use Synopsys library (*.lib*) to obtain input transition time and use standard delay format (*.sdf*) file to obtain nominal gate delay of the switching gate. However, nominal gate delay obtained from *.lib* file and *.sdf* file are different. The percentage difference between two kinds of nominal gate delay is 65.7%. The mismatched delay may causes errors in delay estimation.

Figure 4.3 shows the histogram of *D* and D^* (X axis) for the benchmark circuit *leon3mp* (1M gates). Y axis shows the number of test patterns in every interval. Since the clock period of *leon3mp* is 4 ns, there is at least one path which D^* is longer than clock period in 12366 test patterns (39.7%). These test patterns are called *timing-violation test patterns*.

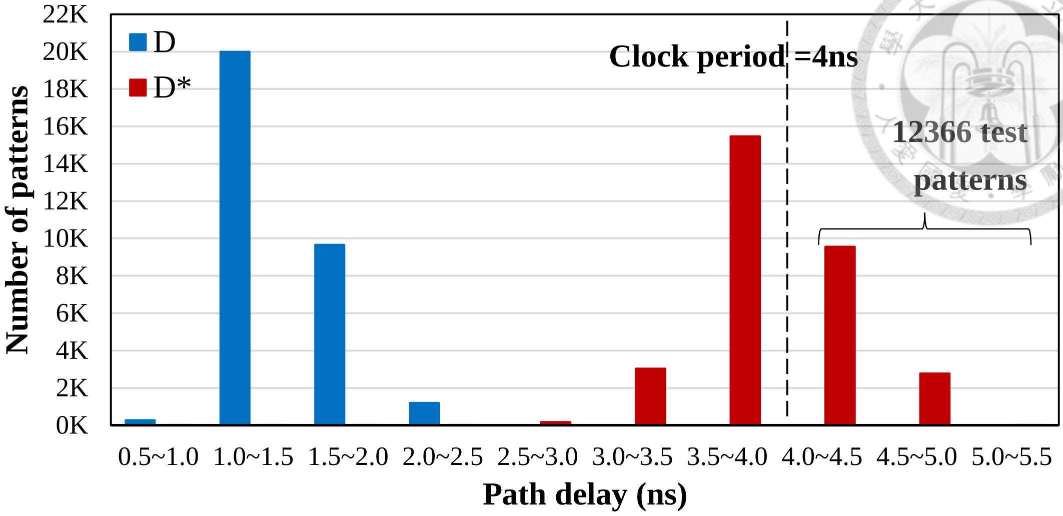


Figure 4.3 Histogram of path delay (leon3mp)

4.3 IR-drop and *Ldi/dt* Experiments

To show the impact of both IR-drop and *Ldi/dt* on extra path delay, we also apply IDEA on a simple package model with multiple clock cycles. The bias voltage variation caused by *Ldi/dt* is added into average PSN estimation.

Figure 4.4 shows the package model, with specific parameter values that is used for simulation. Since the issue of package modeling is difficult, we use a simple RLC circuit as the package model. The benchmark circuits with the package model is solved by KLU.

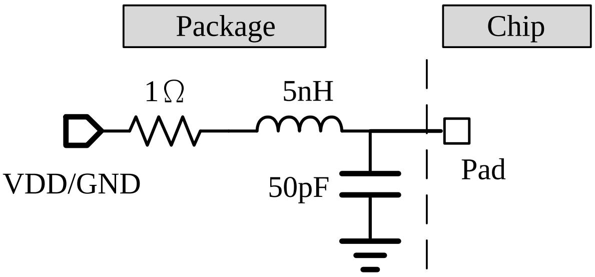


Figure 4.4 Simple package model

TABLE 4.4 shows the error of total path delay for the benchmark circuits with package model. The second and third columns 'E^*_{HSPICE}' show the error of IDEA with respect to HSPICE. The fifth and sixth columns '$E^*_{NANOSIM}$' show the error of IDEA with respect to NANOSIM. These errors are calculated by equation (4.2), where E^* can be replaced by E^*_{HSPICE} and $E^*_{NANOSIM}$. Compared with the error without package model, as shown in TABLE 4.2 and TABLE 4.3, the error with package model is bigger. The reason will be detailed in the Discussion Chapter.

TABLE 4.4 Experimental results of total path delay with package

Circuit	E^*_{HSPICE} (%)	
	Max.	Avg.
s27	132.0	34.9
s208	246.5	53.6
s15850	210.8	25.1
s38417	58.5	22.6
s38584	39.5	9.3
b17	27.8	12.3

Circuit	$E^*_{NANOSIM}$ (%)	
	Max.	Avg.
b18	20.2	18.8
b19	16.9	4.4
leon3mp	23.3	12.8

We use $\Delta D/D$ as *extra delay ratio*. Figure 4.5 shows extra delay ratio with package and without package (Y axis) over 10 clock cycles for *leon3mp*. X axis shows the number of clock cycles. Extra delay ratio without package falls rapidly during the first several clock cycles, and then stabilizes at clock cycle 8 for *leon3mp*. Extra delay ratio with package oscillates, which is caused by resonance effect. There is a wide difference between extra delay ratio with package and without package, especially during the first several clock cycles. Therefore, Figure 4.5 illustrates the need for considering both IR-drop and *Ldi/dt* during timing analysis.

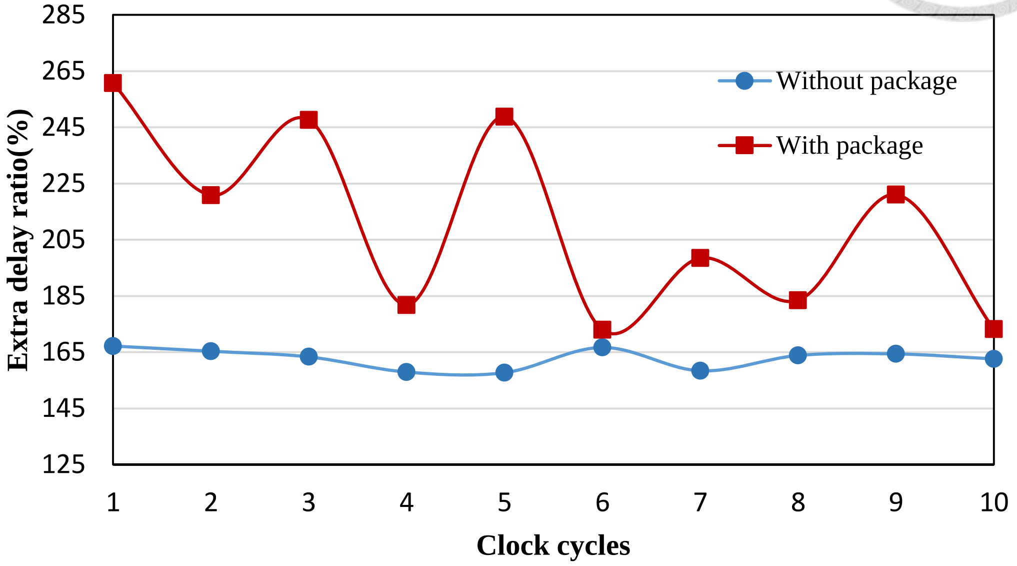


Figure 4.5 Extra delay ratio falls with multiple clock cycles (leon3mp)

4.4 Comparison of Dynamic and Static Window Partition

To evaluate two window partition methods, Figure 4.6 shows the average E^{Δ} (Y axis), where ΔD is simulated with various static number of windows (X axis) of *b17*. If the number of windows is too small, negative E^{Δ} leads to average PSN underestimation. If the number of windows is too many, positive E^{Δ} results in average PSN overestimation. A large number of windows is time consuming.

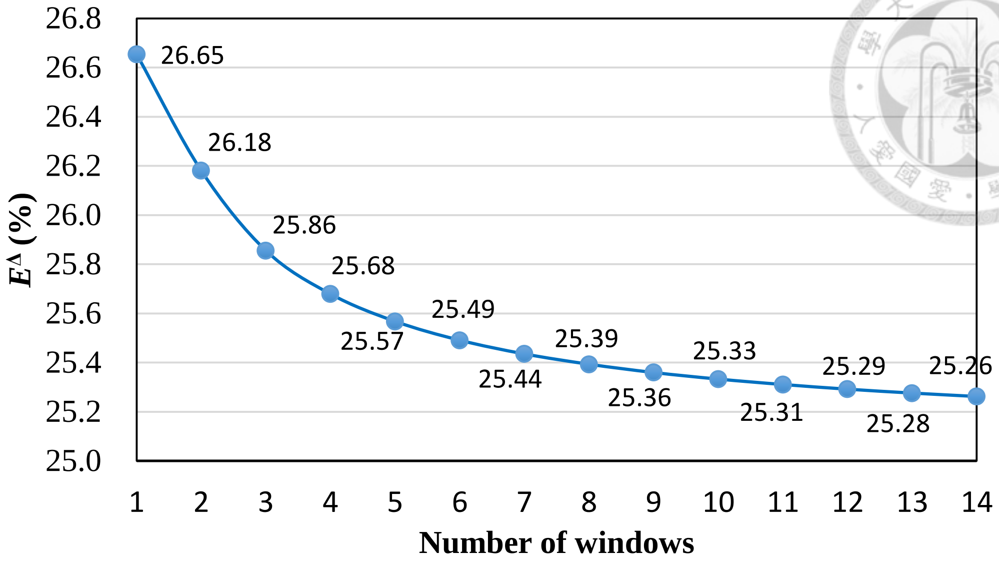


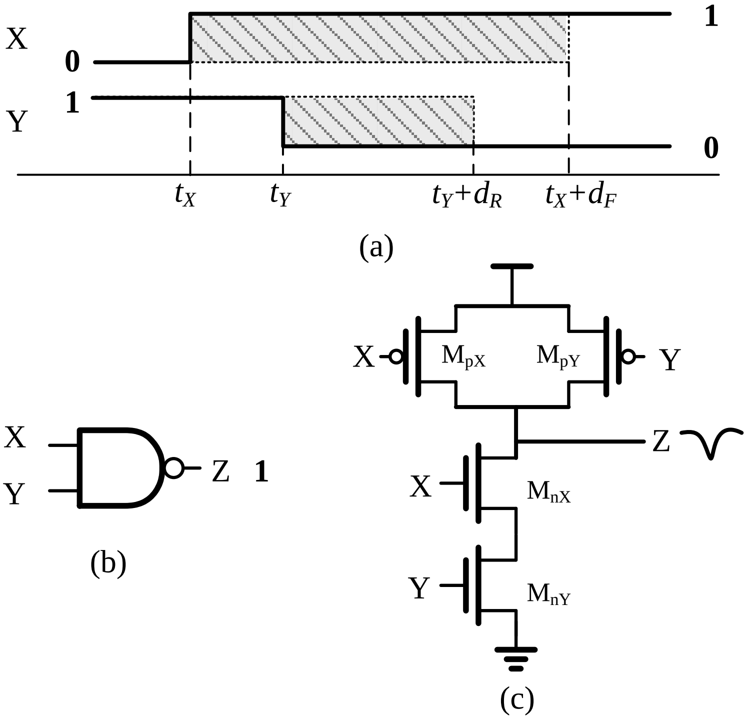
Figure 4.6 Extra path delay error of static window partition (b17)

For *b17*, dynamic window partition separates the whole clock period into three windows in 21% of test set, four windows in 47% of test set, and five windows in 32% of test set. Dynamic window partition results in 25.5% E^{Δ}.

Chapter 5 Discussion

5.1 False Hazard

False hazard happens when gate output transition occurs in HSPICE simulation, but not in timed logic simulation. In Figure 5.1, we use an NAND gate as an illustration example of false hazard. In Figure 5.1(a), input signal *X* is rising at t_X and leads to an output falling condition which will occur at t_X+d_F. Input signal *Y* is falling at t_Y and leads to an output rising condition which will occur at t_Y+d_R. Since t_X is earlier than t_Y and t_X+d_F is later than t_Y+d_R, the capacitor does not discharge completely and start charging. In Figure 5.1(b) and (c), the output signal *Z* in timed logic simulation holds one while there is a glitch in HSPICE simulation.



(a)

(b)

(c)

Figure 5.1 Example of false hazard

To derive a simple model to analyze the charge caused by false hazard, the analysis is divided into three parts. At the beginning, Z=1 since X=0 and Y=1. At t_X, the capacitor stops charging and starts discharging since M_{pX} turns off and M_{nX} turns on. At t_Y, the capacitor stops discharging and starts charging since M_{pY} turns on and M_{nY} turns off. Therefore, we use

$$Q_X = Q \times \frac{t_Y - t_X}{d_X} \tag{5.1}$$

to estimate the charge caused by discharging the capacitor. Q is equal to Q_{IN}+Q_{SW} or Q_{IN}, which is determined by four cases in TABLE 3.1. Q_X is used in average falling current calculation. Since Q_X is equal to the quantity of charge caused by charging the capacitor, we also use Q_X in average rising current calculation.

5.2 Limitation of Multiple Clock Cycles

The reason for the limitation is that we use large window width as a time unit of simulation. Large window width causes inaccuracy in solving equation (5.2). In this equation, window width only affects $\boldsymbol{V}'$ approximation.

$$\boldsymbol{G} \cdot \boldsymbol{V} + \boldsymbol{C} \cdot \boldsymbol{V}' = \boldsymbol{I} \tag{5.2}$$

We use BE method to approximate $\boldsymbol{V}'$, as shown in equation (5.3). In our tool, we use window width as h.

$$\boldsymbol{V}(t+h) = \boldsymbol{V}(t) + h \cdot \boldsymbol{V}'(t+h) \tag{5.3}$$

BE method is based on a truncated Taylor series expansion. Therefore, the truncation leads to the *local truncation error* (LTE) at every time step. To determine LTE for BE method, we expand $\boldsymbol{V}(t+h)$ in the Taylor series and obtain $\boldsymbol{V}_{EX}(t+h)$, as shown in equation (5.4).

$$\boldsymbol{V}_{EX}(t+h)=\boldsymbol{V}(t)+h\cdot\boldsymbol{V}'_{EX}(t+h)-\frac{h^2}{2}\boldsymbol{V}''(t) \tag{5.4}$$

, where $\boldsymbol{V}'_{EX}(t+h)$ is the first derivative of $\boldsymbol{V}_{EX}(t+h)$ with respect to time. We substitute equation (5.3) into equation (5.2) and obtain equation (5.7).

$$\boldsymbol{G}\cdot\boldsymbol{V}(t+h)+\boldsymbol{C}\cdot\frac{\boldsymbol{V}(t+h)-\boldsymbol{V}(t)}{h}=\boldsymbol{I}(t+h) \tag{5.5}$$

$$\left[\boldsymbol{G}+\frac{\boldsymbol{C}}{h}\right]\cdot\boldsymbol{V}(t+h)=\boldsymbol{I}(t+h)+\frac{\boldsymbol{C}}{h}\cdot\boldsymbol{V}(t) \tag{5.6}$$

$$\boldsymbol{V}(t+h)=\left[\boldsymbol{G}+\frac{\boldsymbol{C}}{h}\right]^{-1}\cdot\left[\boldsymbol{I}(t+h)+\frac{\boldsymbol{C}}{h}\cdot\boldsymbol{V}(t)\right] \tag{5.7}$$

Similarly, we substitute equation (5.4) into equation (5.2) and obtain equation (5.10).

$$\boldsymbol{G}\cdot\boldsymbol{V}_{EX}(t+h)+\boldsymbol{C}\cdot\frac{\boldsymbol{V}_{EX}(t+h)-\boldsymbol{V}(t)+\frac{h^2}{2}\cdot\boldsymbol{V}''(t)}{h}=\boldsymbol{I}(t+h) \tag{5.8}$$

$$\left[\boldsymbol{G}+\frac{\boldsymbol{C}}{h}\right]\cdot\boldsymbol{V}_{EX}(t+h)=\boldsymbol{I}(t+h)+\frac{\boldsymbol{C}}{h}\cdot\boldsymbol{V}(t)-\frac{\boldsymbol{C}\cdot h}{2}\cdot\boldsymbol{V}''(t) \tag{5.9}$$

$$\boldsymbol{V}_{EX}(t+h)=\left[\boldsymbol{G}+\frac{\boldsymbol{C}}{h}\right]^{-1}\cdot\left[\boldsymbol{I}(t+h)+\frac{\boldsymbol{C}}{h}\cdot\boldsymbol{V}(t)-\frac{\boldsymbol{C}\cdot h}{2}\cdot\boldsymbol{V}''(t)\right] \tag{5.10}$$

In equation (5.11), $\boldsymbol{V}_{LTE}(t+h)$ is the difference between equation (5.5) and equation (5.8), which is the error of solution vector caused by LTE.

$$\boldsymbol{V}_{LTE}(t+h) = \boldsymbol{V}_{EX}(t+h) - \boldsymbol{V}(t+h)$$

$$= \left[\boldsymbol{G} + \frac{\boldsymbol{C}}{h}\right]^{-1} \cdot \left[-\frac{\boldsymbol{C} \cdot h}{2} \cdot \boldsymbol{V}''(t)\right] \quad (5.11)$$

In Figure 5.2, we use $\boldsymbol{V}_{LTE}(t+h)/\boldsymbol{V}(t+h)$ to show the impact of LTE on solution vector (Y axis). X axis shows the number of windows, which is equal to the number of time steps. For *leon3mp*, there are ten windows in a clock cycle, which means that Figure 5.2 shows the result of ten clock cycles. In this figure, since capacitance in the power grid is very small, $\boldsymbol{V}_{LTE}(t+h)$ without package model is very small. However, capacitance and inductance in the package model is very large. As the number of windows increases, $\boldsymbol{V}_{LTE}(t+h)$ increases. Therefore, the limitation of our tool is that the number of continuous clock cycles is limited.

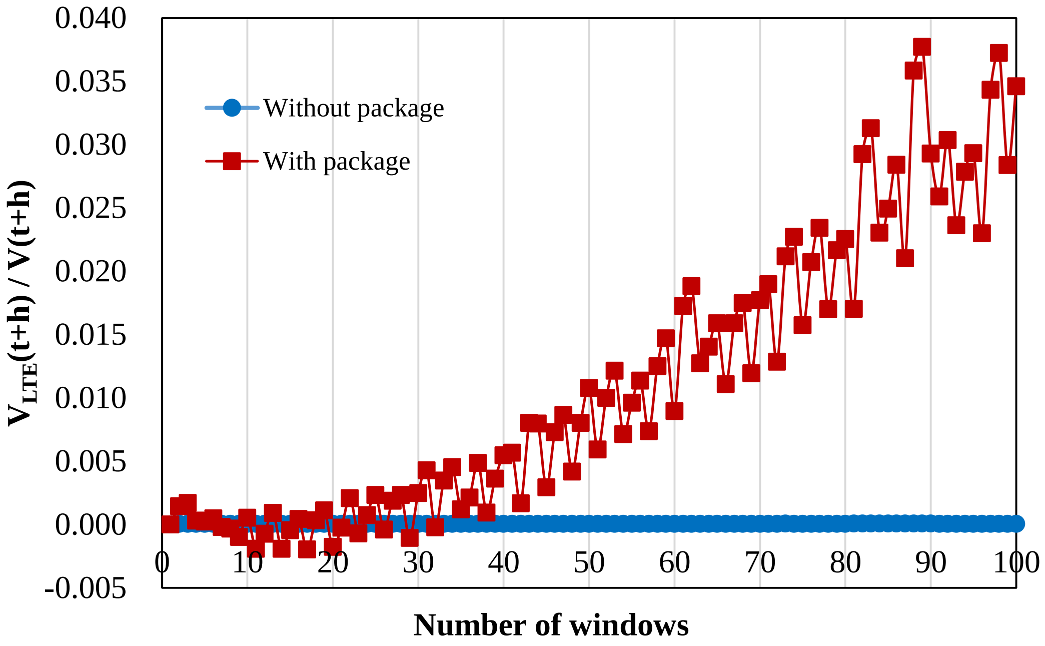


Figure 5.2 Impact of LTE on $\boldsymbol{V}(t+h)$ (leon3mp)

5.3 Interaction between Extra Gate Delay and Event Position

Event position for a switching gate is the timing when input signal of the switching gate changes. Extra gate delay is required to determine event position, which is in turn required to compute extra gate delay. To deal with this interaction, we use nominal gate delay to determine event position for every switching gate. The advantage is short runtime, since event position of every switching gate is determined once for every test pattern. The disadvantage is extra gate delay overestimation. Event positions are delayed owing to extra gate delay, so the number of switching gates occurring in a window may reduce.

We introduce two solutions to deal with the interaction between extra gate delay and event position. The first solution is proposed in [Jiang 2013]. PSN and extra gate delay are calculated for every set of simultaneous switching gates. Event positions of the following switching gates are updated by the calculated extra gate delay. The technique is accurate but slow. A benchmark circuit *p45* (30.6K) is used in [Jiang 2013], so we use a benchmark circuit b17 (32.5K) to make a comparison. The average error of extra path delay is 5%, which is more accurate than our tool (5.43%). However, the runtime is 13 seconds per test pattern, which is 0.08 seconds in our tool. The second solution is based on IDEA, as shown in Figure 5.3.

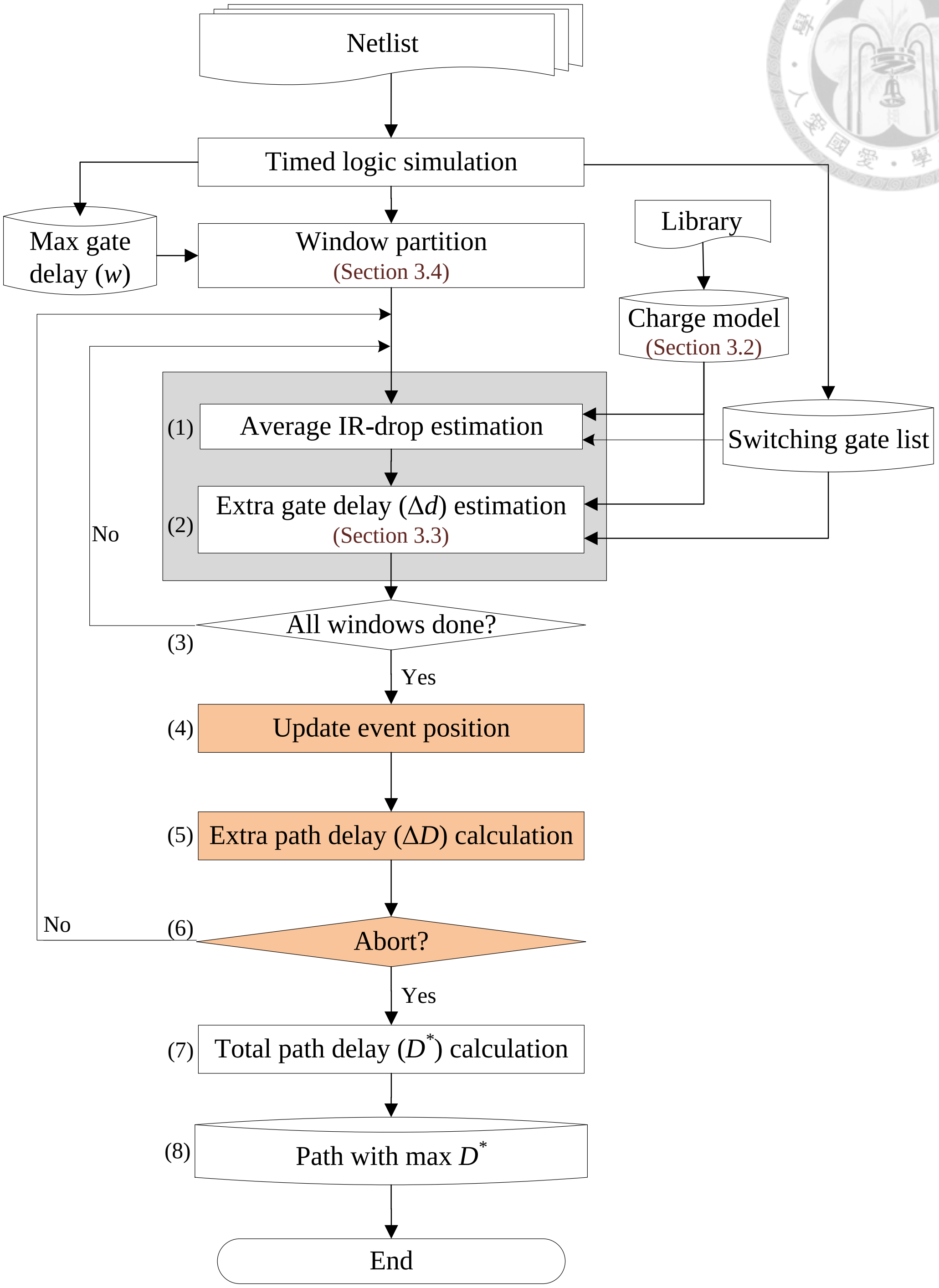


Figure 5.3 IDEA flow with iteration

In this figure, steps 1 to 6 are performed iteratively. There are three steps different from the original IDEA flow: update event position, extra path delay (Δ*D*) calculation and abort (steps 4 to 6).

4) If there is no more windows to process, event position of every switching gate is updated by extra gate delay.

5) Δ*D* calculation is performed to find a path with maximum Δ*D*.

6) If the difference of Δ*D* between this iteration and last iteration is smaller than a user-defined value, move on to step 7; otherwise, repeat steps 1 to 5 with updated event position for the next iteration.

Figure 5.4 shows the change of Δ*D* (Y axis) during twenty iterations for *b17*. The first iteration is the same as original IDEA flow, which uses nominal gate delay to determine event position, so extra gate delay is overestimated. The overestimation may cause the number of switching gates occurring in a window becomes too small, which induces extra gate delay underestimation. Since the overestimation and underestimation, extra path delay oscillates during the first several iterations and then stabilizes after fourteen iterations. At first iteration, average error of extra path delay is 73.5%. After fourteen iterations, the error drops to -3.67%.

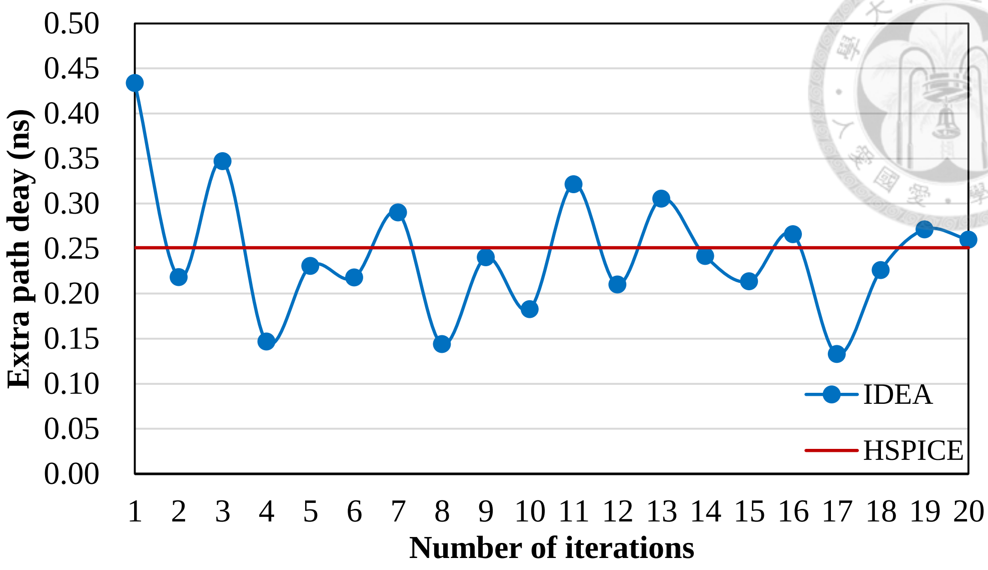


Figure 5.4 Change of extra path delay during twenty iterations (b17)

TABLE 5.1 shows runtime of iterations. The second column ‘NANOSIM’ shows the average runtime of NANOSIM with power grid RC model. The third column ‘1 iteration’ is the same as the results in Chapter 4. The fourth column ’14 iterations’ shows the total runtime of IDEA with fourteen iterations. For small circuit *b17*, our tool achieves four times speed up. For large circuit *leon3mp*, our tool is slower than NANOSIM.

TABLE 5.1 Runtime of iterations

Circuit	Runtime (s/pat)		
	NANOSIM	IDEA	
		1 iteration	14 iterations
b17	2.49	0.08	0.63
leon3mp	61.05	11.40	84.08

5.4 Impact of Different Current Model

We model extra gate delay and extra output transition time as function of charges, but not current model. Therefore, the impact of applying different drain current model is small.

$$\Delta d_{R2} \approx \frac{\Delta\tau_{F1}}{2} + \frac{C\left(\frac{1}{2}VDD - V_{L2}\right)}{\frac{p\times\tau_{F1}}{2\left(V_{H1}-V_{L1}\right)d_{R2}} + \frac{C\left(V_{H2}-V_{L2}\right)}{2d_{R2}}} - \frac{\frac{1}{2}C\times VDD}{\frac{p\times\tau_{F1}}{2\times VDD\times d_{R2}} + \frac{C\times VDD}{2d_{R2}}} + \frac{VDD}{3S_{I2}} - \frac{(V_{H2}-V_{L1})}{3S_{I2}^{*}} \quad (5.12)$$

$$\Delta\tau_{F1} \approx C\times\left(\frac{V_{H1}-V_{L1}}{\frac{p\times\tau_{R0}}{2\left(V_{H0}-V_{L0}\right)d_{F1}} + \frac{C\left(V_{H1}-V_{L1}\right)}{2d_{F1}}} - \frac{VDD-GND}{\frac{p\times\tau_{R0}}{2\times VDD\times d_{F1}} + \frac{C\times VDD}{2d_{F1}}}\right) \quad (5.13)$$

Since we use level-1 current model in extra gate delay estimation, we analyze the impact of more accurate model in this section. We apply another current model on extra gate delay estimation to analyze the impact.

We use equation (5.14) as an example of another drain current model and use Figure 5.5 as an illustration example of rising gate delay estimation. Since the following estimation is similar to the estimation in Section 3.3, we only show the difference caused by the new current model.

$$i_D = \frac{1}{2}\beta\left(v_{GS} - V_{TH}\right)^3 \quad (5.14)$$

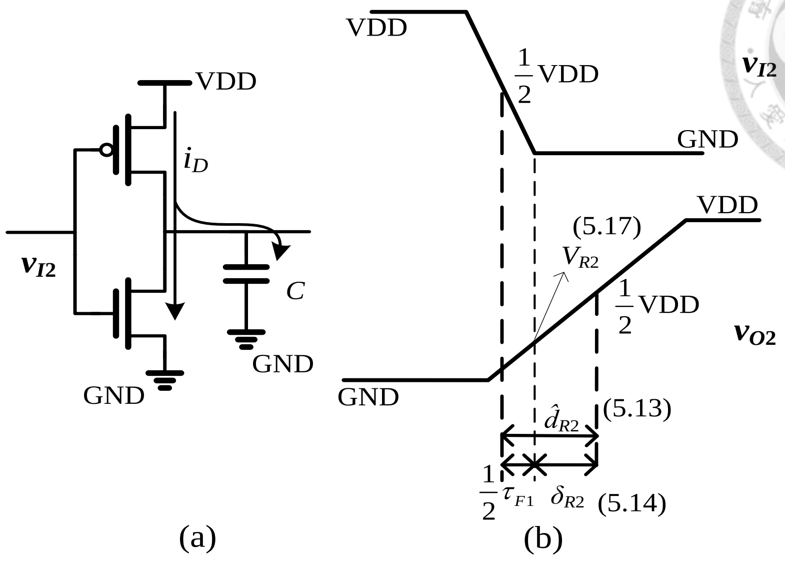


Figure 5.5 Rising gate delay estimation for an inverter

Since v_{GS} changes during input transition, $\hat{d}_{R2}$ estimation is divided into two parts.

$$d_{R2} = \frac{1}{2}\tau_{F1} + \delta_{R2} \tag{5.15}$$

One is the delay before v_{I2} reaches its GND; the other is the delay after v_{I2} reaches GND. The former is equal to half of input transition time of gate 2, which is not influenced by current model. The latter is defined as δ_{R2}, as shown in equation (5.16).

$$\begin{aligned}\delta_{R2} &\approx \frac{C \times \Delta v_{O2}}{i_D} \\ &= \frac{C(\frac{1}{2}VDD - V_{R2})}{\frac{1}{2}\beta(VDD - V_{TH})^3}\end{aligned} \tag{5.16}$$

The derivation of V_{R2}, which is the output voltage when v_{I2} reaches GND, is shown in equations (5.17) to (5.19).

$$C\frac{dv_{O2}}{dt} \approx i_D \tag{5.17}$$

$$C\int_{GND}^{V_{R2}} dv_{O2} \approx \frac{1}{2}\beta\int_{0}^{\tau_{F1}} (S_{I2}\times t - V_{TH})^3 dt \tag{5.18}$$

$$V_{R2} \approx \frac{\beta}{8\times C\times S_{I2}}(VDD - V_{TH})^4 \tag{5.19}$$

We use similar way to calculate $\hat{d}_{R2}^{*}$.

$$d_{R2}^{*} \approx \frac{1}{2}(\tau_{F1} + \Delta\tau_{F1}) + \frac{C(\frac{1}{2}VDD - V_{L2})}{\frac{1}{2}\beta(V_{H2} - V_{L1} - V_{TH})^3} - \frac{(V_{H2} - V_{L1} - V_{TH})}{4S_{I2}^{*}} \tag{5.20}$$

Then, Δd_{R2} can be obtained by

$$\begin{aligned}\Delta d_{R2} \approx{}& \frac{\Delta\tau_{F1}}{2} + \frac{C(\frac{1}{2}VDD - V_{L2})}{\frac{1}{2}\beta(V_{H2} - V_{L1} - V_{TH})^3} - \frac{\frac{1}{2}C\times VDD}{\frac{1}{2}\beta(VDD - V_{TH})^3}\\ &+ \frac{\left(VDD - V_{TH}\right)}{4S_{I2}} - \frac{(V_{H2} - V_{L1} - V_{TH})}{4S_{I2}^{*}}\end{aligned} \tag{5.21}$$

Since S_{I2} and S_{I2}^{*} are very large, we modify equation (5.21) into equation (5.22).

$$\Delta d_{R2} \approx \frac{\Delta\tau_{F1}}{2} + \frac{C(\frac{1}{2}VDD - V_{L2})}{\frac{1}{2}\beta(V_{H2} - V_{L1} - V_{TH})^3} - \frac{\frac{1}{2}C\times VDD}{\frac{1}{2}\beta(VDD - V_{TH})^3} + \frac{VDD}{4S_{I2}} - \frac{(V_{H2} - V_{L1})}{4S_{I2}^{*}} \tag{5.22}$$

The values of β and V_{TH} are not determined yet, so we use peak current to replace the current in equation (5.22).

$$\Delta d_{R2} \approx \frac{\Delta\tau_{F1}}{2} + \frac{C\left(\frac{1}{2}VDD - V_{L2}\right)}{\tilde{I}_{PR2}^{*}} - \frac{\frac{1}{2}C\times VDD}{\tilde{I}_{PR2}} + \frac{VDD}{4S_{I2}} - \frac{(V_{H2} - V_{L1})}{4S_{I2}^{*}} \tag{5.23}$$

In equation (5.23), four terms are influenced by different current model: $\tilde{I}_{PR2}$, $\tilde{I}^{*}_{PR2}$, $4S_{I2}$ and $4S^{*}_{I2}$. First, we focus on peak current. Equation (5.24) shows the integral of i_D with the assumption of $d_{R2} \gg 0.5\tau_{F1}$.

$$Q_D = \int_0^{d_{R2}} \left[\frac{\beta}{2} \left(v_{GS} - V_{TH} \right)^3 \right] dt$$

$$= \int_0^{\frac{1}{2}\tau_{F1}} \left[\frac{\beta}{2} \left(v_{GS} - V_{TH} \right)^3 \right] dt + \int_{\frac{1}{2}\tau_{F1}}^{d_{R2}} \left[\frac{\beta}{2} \left(VDD - V_{TH} \right)^3 \right] dt$$

$$\approx d_{R2} \times \frac{\beta}{2} \left(VDD - V_{TH} \right)^3 = Q_D \qquad (5.24)$$

$$\frac{Q_D}{w} \approx \frac{Q_D}{w} = \frac{Q_{SW} + Q_{IN}}{2w} = \frac{1}{2} \bar{I}_{PR2} \qquad (5.25)$$

We substitute equation (5.24) into equation (5.25) and obtain

$$\frac{d_{R2}}{w} \times \frac{\beta}{2} \left(VDD - V_{TH} \right)^2 \approx \frac{1}{2} \bar{I}_{PR2} \qquad (5.26)$$

Therefore, $\tilde{I}_{PR2}$ can be calculated by

$$\tilde{I}_{PR2} = \bar{I}_{PR2} \times \frac{w}{2d_{R2}} = \frac{p \times \tau_{F1}}{2 \times VDD \times d_{R2}} + \frac{C \times VDD}{2 \times d_{R2}} \qquad (5.27)$$

Similarly, $\tilde{I}^{*}_{PR2}$ can be obtained by

$$\tilde{I}^{*}_{PR2} = \frac{p \times \tau_{F1}}{2\left(V_{H1} - V_{L1} \right) d_{R2}} + \frac{C\left(V_{H2} - V_{L2} \right)}{2d_{R2}} \qquad (5.28)$$

From equations (5.24) to (5.28), the calculation of $\tilde{I}_{PR2}$ and $\tilde{I}^{*}_{PR2}$ is not significantly influenced by different current model as long as the assumption of $d_{R2} \gg 0.5\tau_{F1}$ is valid.

Second, we deal with $4S_{I2}$ and $4S_{I2}^{*}$, which are in the denominator of equation (5.23). Since S_{I2} and S_{I2}^{*} are very large, we can make this approximation:

$$\frac{VDD}{3S_{I2}}-\frac{V_{H2}-V_{L1}}{3S_{I2}^{*}}\approx\frac{VDD}{4S_{I2}}-\frac{V_{H2}-V_{L1}}{4S_{I2}^{*}}\approx 0 \qquad (5.29)$$

Therefore, the impact of different drain current model on extra gate delay estimation is very small.

Chapter 6 Conclusion and Future Work

This thesis proposes an efficient and accurate PSN-aware dynamic timing analyzer, IDEA, which considers both IR-drop and *Ldi/dt*. IDEA uses window partition to calculate average PSN in a window so that we can find good balance between accuracy and runtime. IDEA is very scalable because the gate delay is modeled as a function of charges. Therefore, IDEA estimates gate delay accurately without SPICE simulation for each logic gate. The experimental results show, for small circuits, the average error of total path delay is less than 1% compared with HSPICE. For large circuits, we achieved eight times speed up compared with NANOSIM.

After performing IDEA on a 1M gate benchmark circuit, experimental results show that 369 timing-violation test patterns (out of 31K test patterns) are identified. A test pattern modification is needed by these test patterns to prevent timing failure and avoid yield loss. Previous research papers about test pattern modification do not handle timing-violation well since they do not have good techniques to translate PSN to extra gate delay. Existing techniques modify test patterns to minimize power for critical paths [Wen 2007][Enokimoto 2009][Miyase 2011]. X-filling is used to reduce switching activity at neighboring logic gates near critical paths [Wen 2007][Miyase 2011]. *Clock-gating* and *FF-silencing* are applied on flip-flops, which are in the fan-in cone of neighboring logic gates near critical paths [Enokimoto 2009]. However, there are two problems in these previous research papers: (1) how to determine the range of neighboring logic gates and (2) how to guarantee timing-safety after test pattern modification. We use Figure 6.1 to illustrate the first problem. Logic gates in critical area (radius *R*) are neighboring logic gates. The value of *R* is hard to determine since it is unwarrantable that the impact of logic gates outside the critical area can be ignored. For the second problem, these techniques only reduce power consumption without considering timing. Therefore, the test patterns are power-safety after test pattern modification, but not always timing-safety.

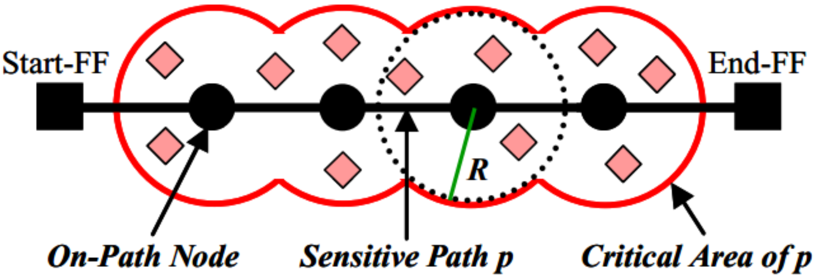


Figure 6.1 Neighboring logic gates near critical path [Enokimoto 2009]

Nowadays, by means of IDEA, we can obtain PSN-induced extra gate delay accurately and efficiently. Once we develop a novel tool, which modifies timing-violation test patterns without test length inflation and fault coverage loss, we can obtain a timing-safety test set even for large circuits with lots of test patterns.

References

[Ahmadi 2003] T. Ahmadi and F. Najm, “Timing Analysis in Presence of Power Supply Noise and Ground Voltage Variations,” *Proc. IEEE Int. Conf. Comput.-Aided Design*, 2003, pp. 1–8.

[Ahmed 2007] N. Ahmed, M. Tehranipoor, and V. Jayaram, “Transition Delay Fault Test Pattern Generation Considering Supply Voltage Noise in a SOC Design,” *Proc. of Design Automation Conf.*, 2007, pp. 533-538.

[Apache 2011] Apache RedHawk User Manual, 2011.

[Aparicio 2012] M. Aparicio, M. Comte, F. Azais, Y. Bertrand, M. Renovell, J. Jiang, I. Polian and B. Becker, “An IR-Drop Simulation Principle Oriented to Delay Testing,” *27th Conference on Design of Circuits and Integrated Systems (DCIS)*, Avignon, France, 2012.

[Aparicio 2013] M. Aparicio, M. Comte, F. Azaïs, M. Renovell, J. Jiang, I. Polian, B. Becker, “Pre-characterization Procedure for a Mixed Mode Simulation of IR-Drop Induced Delays,” *Proc. of IEEE LATW*, 2013.

[Cadence 2009] Package effect on chip power supply: can designers afford to ignore it? https://www.cadence.com/rl/Resources/conference_papers/9.3Presentation.pdf

[Chen 1997] H. Chen and D. Ling, "Power supply noise analysis methodology for deep submicron VLSI design," *Proceedings of ACM/IEEE Design Automation Conf.*, pp. 638–643, 1997.

[Chen 1998] H. H. Chen and J. S. Neely, "Interconnect and circuit modeling techniques for full-chip power noise analysis," *Trans. on ComponentsPackaging II,* 1998.

[Chen 2012] Q. Chen, S.-H. Weng and C.-K. Cheng, "A Practical Regularization Technique for Modified Nodal Analysis in Large-Scale Time-Domain Circuit Simulation," *IEEE Trans. on Computer-Aided Design*, 31(7):1031-1040, 2012.

[Davis 2010] T. A. Davis and E. P. Natarajan, "Algorithm 907: KLU, a direct sparse solver for circuit simulation problems," *ACM Trans. MS*, vol.37, no.3, 2010.

[Ding 2013] W.-S. Ding, H.-Y. Hsieh, and J. C.-M. Li, "Test Pattern Modification for IR-drop Reduction," *IEEE Int'l Test Conf. poster*, 2013.

[Devanathan 2007] V. R. Devanathan, C. P. Ravikumar, and V. Kamakoti. "Glitch-aware pattern generation and optimization framework for power-safe scan test," *Proc. VLSI Test Symp.*, pages 167–172, 2007.

[Enami 2008] T. Enami, S. Ninomiya, and M. Hashimoto, "Statistical timing analysis considering spatially and temporally correlated dynamic power supply noise," *Proc. of ISPD*, pp.160-167, 2008.

[Enokimoto 2009] K. Enokimoto, X. Wen, Y. Yamato, K. Miyase, H. Sone, S. Kajihara, M. Aso, and H. Furukawa, "CAT: A Critical-Area-Targeted Test Set Modification Scheme for Reducing Launch Switching Activity in At-Speed Scan Testing," *Proc. of Asian Test Symposium*, Nov. 2009, pp. 99-104.

[Girard 2002] P. Girard, "Survey of Low-Power Testing of VLSI Circuits," *Design & Test of Computers*, 2002, vol. 19, issue 3, pp. 80-90.

[Hashimoto 2004] M. Hashimoto, J. Yamaguchi and H. Onodera, "Timing Analysis Considering Spatial Power/Ground Level Variation," *International Conference on Computer-Aided Design,* 2004.

[Hashimoto 2008] M. Hashimoto, J. Yamaguchi, T. Sato, and H. Onodera, "Timing Analysis Considering Temporal Supply Voltage Fluctuation," *Trans. on Information and Systems*, vol. E91-D, issue 3, 2008, pp. 655-660.

[Hedenstiema 1987] N. Hedenstiema and K. 0. Jeppson. CMOS circuit speed and buffer optimization. *IEEE Trans. on Computer-Aided Design*, 6(2):270-281, March 1987.

[Jiang 1999] Y. M. Jiang and K. T. Cheng, "Analysis of Performance Impact Caused by

Power Supply Noise in Deep Submicron Devices", *Proceedings of ACM/IEEE Design Automation Conf.*, pp.760-765, 1999.

[Jiang 2013] J. Jiang, M. Aparicio, M. Comte, F. Aza¨ıs, M. Renovell and I. Polian, "MIRID: Mixed-Mode IR-Drop Induced Delay Simulator," *Proc. of Asian Test Symposium*, Nov. 2013, pp. 177-182.

[Li 2013] Y.-H. Li, W.-C. Lien, I.-C. Lin, K.-J. Lee, "Capture-Power-Safe Test Pattern Determination for At-Speed Scan-Based Testing," *IEEE Trans. on Computer-Aided Design*, vol. 33, No. 1, 127-138, 2013.

[Lin 2008] H.-T. Lin and J. C-M. Li, "Simultaneous Capture and Shift Power Reduction Test Pattern Generator for Scan Testing," *IET Computers and Digital Techniques*, 2008, vol. 2, issue 2, pp.132-141.

[Liou 2000] J.-J. Liou, A. Krstić, Y.-M. Jiang and K.-T. Cheng, "Path selection and pattern generation for dynamic timing analysis considering power supply noise effects," *IEEE International Conference on Computer Aided Design*, pp. 493-497, 2000.

[Liou 2003] J.-J. Liou, A. Krstic, Y.-M. Jiang and K.-T. Cheng, ‘‘Modeling, Testing, and Analysis for Delay Defects and Noise Effects in Deep Submicron Devices,’’ *IEEE Trans. Computer-Aided Design*, vol. 22, no. 6, pp. 756-769, 2003.

[Ma 2009] J. Ma, J. Lee, and M. Tehranipoor, "Layout-Aware Pattern Generation for Maximizing Supply Noise Effects on Critical Paths," *Proc. of VLSI Test Symposium*, 2009, pp. 221-226.

[Ma 2011] J. Ma, M. Tehranipoor, "Layout-Aware Critical Path Delay Test Under Maximum Power Supply Noise Effects," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol.30, no.12, pp.1923-1934, Dec. 2011.

[Maurine 2001] P. Maurine, M. Rezzoug, D. Auvergne "Output transition time modeling of CMOS structures," *IEEE Int. Symp. on Circuits and Syst.*, vol. 5, pp. 363-366, 2001.

[Miyase 2011] K. Miyase, X. Wen, M. Aso, H. Furukawa, Y. Yamato, and S. Kajihara, "Transition-Time-Relation based capture-safety checking for at-speed scan test generation," *Proc. Design Automation and Test in Europe Conf.*, pp. 895-898, 2011.

[Najm 2010] F. N. Najm, "*Circuit Simulation*," New York: Wiley, 2010.

[Nassif 2000] S. R. Nassif and J. N. Kozhaya, "Fast Power Grid Simulation," *IEEE Design Automation Conference*, pp. 156-161, Jun. 2000.

[Ogasahara 2007] Y. Ogasahara, T. Enami, M. Hashimoto, T. Sato, and T. Onoye, "Validation of a Full-Chip Simulation Model for Supply Noise and Delay Dependence on Average Voltage Drop With On-Chip Delay Measurement," *Trans. on Circuits and Systems II: Express Briefs*, vol. 54, issue: 10, 2007, pp.868-872.

[Okumura 2010] T. Okumura, F. Minami, K. Shimazaki, K. Kuwada, M. Hashimoto, "Gate Delay Estimation in STA under Dynamic Power Supply Noise," *Proc. ASPDAC.*, 2010.

[Pant 2003] S. Pant, D. Blaauw, V. Zolotov, S. Sundareswaran and R. Panda, "Vectorless analysis of supply noise induced delay variation," *IEEE International Conference on Computer Aided Design*, pp. 184-191, Nov. 2003.

[Peng 2010] K. Peng, Y. Huang, P. Mallick, W. Cheng, M. Tehranipoor, "Full-Circuit SPICE Simulation Based Validation of Dynamic Delay Estimation," *Proc. Eur. Test Symp.*, 2010.

[Rao 2012] S. K. Rao, C. Sathyanarayana, A. Kallianpur, R. Robucci, C. Patel, "Estimating Power Supply Noise and Its Impact on Path Delay," *VLSI Test Symposium*, 2012.

[Remersaro 2006] S. Remersaro, X. Lin, Z. Zhang, S. M. Reddy, I. Pomeranz, and J. Rajski, "Preferred Fill: A Scalable Method to Reduce Capture Power for Scan Based Designs," *Proc. of Int'l Test Conf.*, Paper 32.2, 2006, pp. 1-10.

[Saint-Laurent 2004] M. Saint-Laurent, and M. Swaminathan, "Impact of power-supply noise on timing in high-frequency microprocessors," *Trans. on Advanced Packaging*, vol. 27, issue 1, 2004, pp. 135-144.

[Saleh 2000] R. Saleh, S. Z. Hussain, S. Rochel, and D. Overhauser, "Clock skew verification in the presence of IR-drop in the power distribution network," *IEEE Trans. on Computer-Aided Design, vol. 19, No. 6*, pp. 635–644, 2000.

[Saxena 2003] J. Saxena, K. M. Butler, V.B. Jayaram, S. Kundu, N.V. Arvind, P. Sreeprakash, and M. Hachinger, "A Case Study of IR-drop in Structured At-Speed Testing," *Proc. of Int'l Test Conf.*, vol.1, 2003, pp. 1098-1104.

[Shepard 1996] K. L. Shepard and V. Narayanan, "Noise in deep submicron digital design," *Proceedings of IEEE ICCAD*, pp. 524–531, 1996.

[Synopsys 2008] Synopsys library compiler user guide, 2008.

[Tehranipoor 2010] M. Tehranipoor and K.M. Butler, "Power Supply Noise: A Survey on Effects and Research," *IEEE Design & Test of Computers*, vol.27, issue 2, 2010, pp. 51-67.

[Todri 2012] A. Todri, A. Bosio, L. Dilillo, P. Girard, and A. Virazel, "Uncorrelated Power Supply Noise and Ground Bounce Consideration for Test Pattern Generation," *Trans. on Very Large Scale Integration*, vol. 21, issue 5, 2012.

[Varma 2012] P. Varma, "Current and Future Directions in Automatic Test Pattern Generation for Power Delivery Network Validation," *Proc. of Asian Test Symposium*, Nov. 2012, pp. 233-238.

[Wang 2005] J. Wang, X. Lu, W. Qiu, Z. Yue, S. Fancler, W. Shi and D. M. H. Walker, "Static Compaction of Delay Tests Considering Power Supply Noise," *Proc. of VLSI Test Symposium*, Palm Springs, CA, May 2005, pp. 235-240.

[Wang 2006] J. Wang, D. M. H. Walker, A. Majhi, B. Kruseman, G. Gronthoud, L. E. Villagra, P. v. d. Wiel, and S. Eichenberger, "Power supply noise in delay testing," *Proc. of Int'l Test Conf.*, 2006, pp. 1-10.

[Wang 2007] J. Wang, D. M. Walker, X. Lu, A. Majhi, B. Kruseman, G. Gronthoud, L. E. Villagra, P. v. d. Wiel, and S. Eichenberger, "Modeling Power Supply Noise in Delay Testing," *IEEE Design & Test of Computers*, vol.24, issue 3, 2007, pp. 226-234.

[Wen 2005] X. Wen, Y. Yamashita, S. Morishima S. Kajihara, L.T. Wang, K. K. Saluja, and K. Kinoshita, "Low-Capture-Power Test Generation for Scan-Based At-Speeding Testing," *Proc. of Int'l Test Conf.*, 2005, pp. -1028.

[Wen 2007] X. Wen, K. Miyase, T. Suzuki, S. Kajihara, Y. Ohsumi, and K. K. Saluja, “Critical-path-aware X-filling for Effective IR-drop Reduction in At-speed Scan Testing,” *Proc. of Design Automation Conf.*, 2007, pp. 527-532.

[Wen 2008] X. Wen et al. “A capture-safe test generation scheme for at-speed scan testing,” *Proc. European Test Symp.*, pages 55–60, 2008.

[Wu 2010] M.-F. Wu, H.-C. Pan, T.-H. Wang, J.-l. Huang et al. “Improved weight assignment for logic switching activity during at-speed test pattern generation,” *ASP-DAC*, 2010.

[Zhu 2003] Z. Zhu, B. Yao and C.-K. Cheng, “Power network analysis using an adaptive algebraic multigrid approach,” *IEEE Design Automation Conference*, pp. 105-108, jun. 2003.