POWER SUPPLY NOISE IN DELAY TESTING

A Dissertation

by

JING WANG

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

DOCTOR OF PHILOSOPHY

August 2007

Major Subject: Computer Engineering

POWER SUPPLY NOISE IN DELAY TESTING

A Dissertation

by

JING WANG

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

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	Duncan M. Walker
Committee Members,	Rabinarayan Mahapatra
	Vivek Sarin
	Weiping Shi
Head of Department,	Valerie E. Taylor

August 2007

Major Subject: Computer Engineering

ABSTRACT

Power Supply Noise in Delay Testing.

(August 2007)

Jing Wang, B.S., Zhejiang University, China

Chair of Advisory Committee: Dr. Duncan M. Walker

As technology scales into the Deep Sub-Micron (DSM) regime, circuit designs have become more and more sensitive to power supply noise. Excessive noise can significantly affect the timing performance of DSM designs and cause non-trivial additional delay. In delay test generation, test compaction and test fill techniques can produce excessive power supply noise. This will eventually result in delay test overkill.

To reduce this overkill, we propose a low-cost pattern-dependent approach to analyze noise-induced delay variation for each delay test pattern applied to the design. Two noise models have been proposed to address array bond and wire bond power supply networks, and they are experimentally validated and compared. Delay model is then applied to calculate path delay under noise. This analysis approach can be integrated into static test compaction or test fill tools to control supply noise level of delay tests. We also propose an algorithm to predict transition count of a circuit, which can be applied to control switching activity during dynamic compaction.

Experiments have been performed on ISCAS89 benchmark circuits. Results show that compacted delay test patterns generated by our compaction tool can meet a moderate noise or delay constraint with only a small increase in compacted test set size. Take the

benchmark circuit s38417 for example: a 10% delay increase constraint only results in 1.6% increase in compacted test set size in our experiments. In addition, different test fill techniques have a significant impact on path delay. In our work, a test fill tool with supply noise analysis has been developed to compare several test fill techniques, and results show that the test fill strategy significant affect switching activity, power supply noise and delay. For instance, patterns with minimum transition fill produce less noise-induced delay than random fill. Silicon results also show that test patterns filled in different ways can cause as much as 14% delay variation on target paths. In conclusion, we must take noise into consideration when delay test patterns are generated.

DEDICATION

To my parents:

without their support, this would not have been possible.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor, Dr. Duncan M. (Hank) Walker, for his guidance, patience and continuous support throughout my doctoral study. I highly appreciate his contributions to my intellectual and personal development in the past five years. I would also like to thank him for guiding me in my dissertation research with such dedication and consideration, and never failing to pay attention to any details of my work. His technical insight, his novel ideas and his encouragement are all essential to this work. This dissertation would never have been accomplished without his technical and editorial advice.

I would like to extend my gratefulness to my committee: Dr. Weiping Shi, Dr. Vivek Sarin and Dr. Rabinarayan Mahapatra. They provided a lot of valuable suggestions and personal encouragement, and I learned a lot from talking to them.

A special thanks to Dr. Shi and his student Xiang Lu for their contributions to my modeling theories and to the experiment set-ups. I highly appreciate the time and effort Xiang spent in generating circuit layouts and in executing circuit simulations. He helped me pave a path through bushes during my early stage of research.

My gratitude also goes to my industry mentors and colleagues in the former Philips Research Lab, which now becomes part of NXP Semiconductors. Dr. Ananta Majhi supervised me during my internship in Philips, and he worked together with me on every problem in my research and paid attention to every detail in the work. Dr. Bram Kruseman helped me collect silicon data on a Philips design and edited our publications. Guido Gronthoud taught me to use tools for carrying out experiments. Paul van de Wiel spent

quite some time in discussion with me and provided solutions to a critical problem in my research. Dr. Stefan Eichenberger and Luis Elvira Villagra gave me a lot of insight comments on the research. I had a wonderful internship experience working with all my colleagues there, and I also very much enjoyed my stay in this beautiful country.

Thanks to my teammates Wangqi Qiu, Bin Xue, Sagar Sabade, Abhijit Prasad, Lei Wu and Ziding Yue. I learned a lot from them in the past several years. Especially I want to thank Wangqi Qiu for his help and advice on my research. He gave me many valuable suggestions when I made a decision on the research topic. Another special thanks goes to Ziding Yue for her important contribution to the experiment part of my research.

My research was funded by Semiconductor Research Corporation (SRC) and National Science Foundation (NSF). I thank these sponsors for providing financial support.

Finally, I want to acknowledge the love and support of my parents. They were always there for me whenever I had problems, and they always shared my happiness for every progress I made. I'm deeply indebted to them, more than my words can ever express.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

Page

Page

Page

LIST OF TABLES

Page

1. INTRODUCTION

As technology advances into the deep submicron (DSM) regime, designs are becoming increasingly sensitive to power supply noise. Excessive noise can cause performance degradation and signal integrity problems. Moreover, it can significantly affect the timing performance of DSM designs.

Power supply noise refers to the noise on the supply and ground network, which reduces device voltage levels and increases signal delay. As operating frequency and gate density increase, power density increases as well. Meanwhile, DSM CMOS technologies require the use of reduced supply voltages. These technology trends have led to higher current density, and consequently increased power supply noise.

Data from the International Technology Roadmap for Semiconductors (ITRS) report [1], is projected in the following four figures. Figure 1 shows the on-chip operating frequency data based on the fundamental transistor delay and an assumed maximum number of 12 inverter delays beginning 2007; after 2007, the fundamental reduction rate is modeled as ~ -14.7% for the transistor delay and results in a ~17.2% growth trend of the on-chip frequency through 2020.

Figure 2 plots transistor density data for cost-performance MPU and high-performance MPU and ASIC product generations in the unit of million transistors per square centimeter. The transistor density for Cost-Performance MPU includes logic

This dissertation follows the style and format of *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*.

only, while the transistor density for high-performance MPU includes on-chip SRAM as well. Both increase exponentially with time.

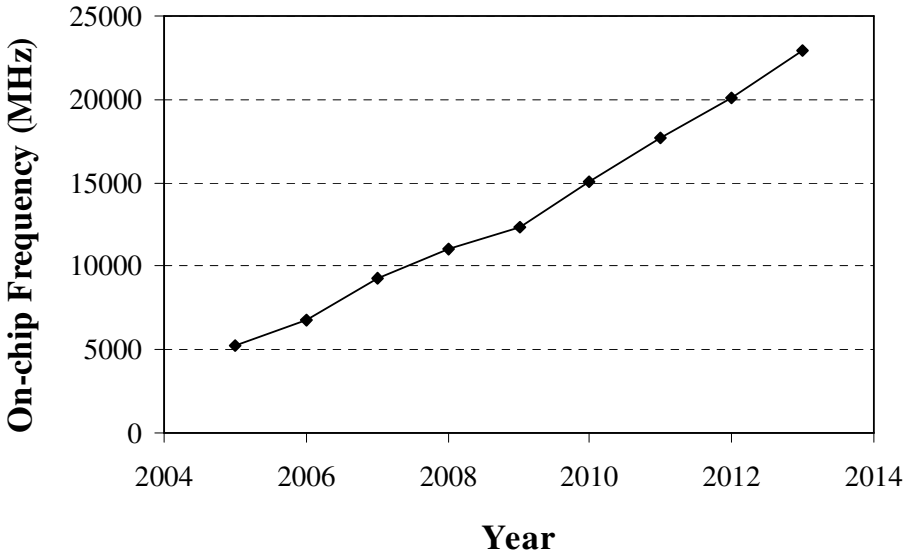

Figure 1. On-chip frequency in the near-term years.

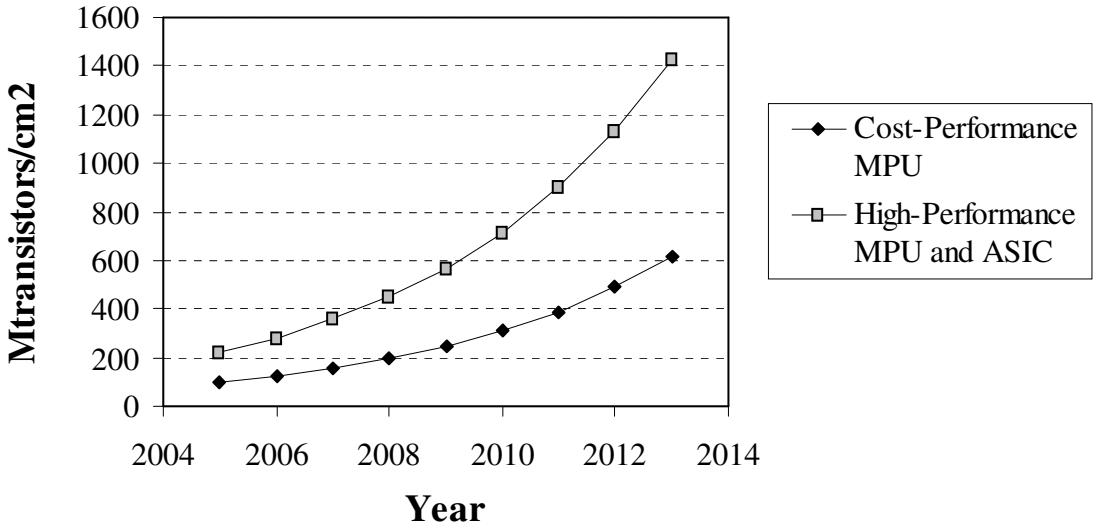

Figure 2. Transistor density in the near-term years.

Figure 3 shows the increase in maximum power density for both cost-performance and high-performance MPU products for maximum power calculation. We can see that the maximum power density for high-performance MPU products will become stable after year 2008. However, the power density for cost-performance MPU products keeps increasing, and will continue to increase after year 2013, which is not shown here.

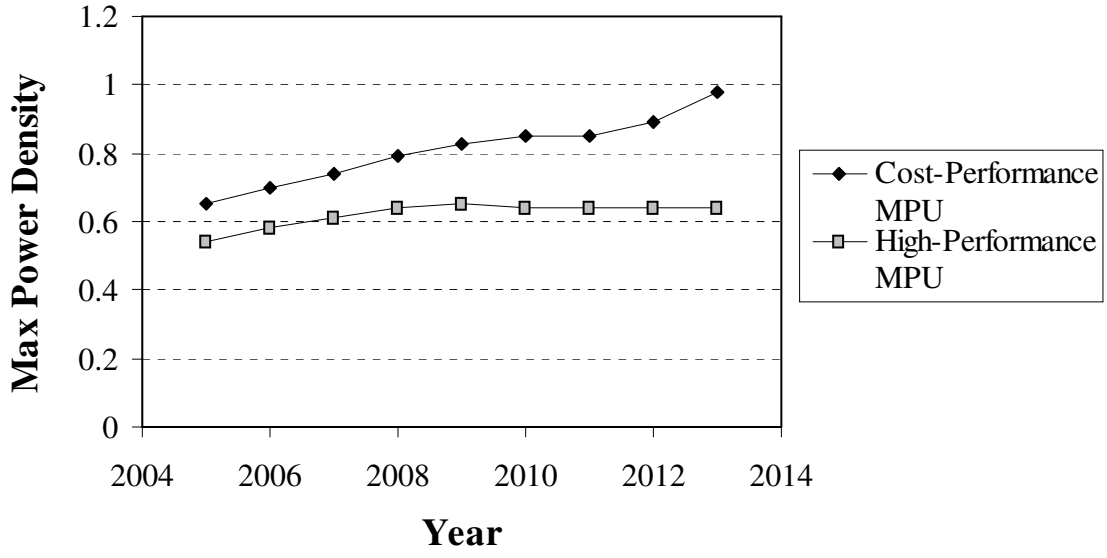

Figure 3. Power density in the near-term years.

Figure 4 shows the decreasing supply voltage level for both high-performance and low-power designs.

All the technology trends illustrated in Figure 1 to Figure 4 imply a higher supply noise as technology advances. Furthermore, industry data shows gate delay becoming increasingly sensitive to supply voltage variations due to reduced gate overdrive. In 130 nm CMOS technology, a 10% variation in supply voltage causes a 30% delay variation for typical gates [2]. In 90 nm 0.9 V technology, a 1% voltage change causes

approximately a 4% change in gate delay [3]. This increased sensitivity contributes to a larger power noise impact on delay.

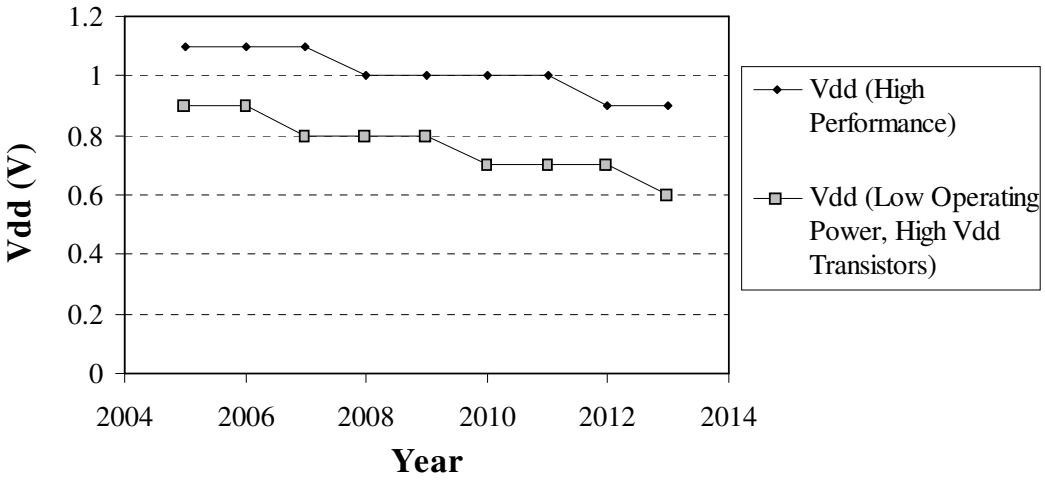

Figure 4. Power supply voltage in the near-term years.

At-speed delay test has been investigated for many years to detect small manufacturing defects that do not cause functional failures, but reduce the speed of circuits. As the noise impact on delay becomes more and more significant, it also inevitably becomes a concern for delay test generation.

In industry, random fill of don't care bits is usually applied to delay test patterns to increase fortuitous detection of non-target defects. Unfortunately, random fill can produce excessive supply noise and result in delay test overkill. Worse, test compaction itself can generate excessive supply noise. Both test fill and test compaction are necessary steps in delay test generation. Therefore, the goal of this work is to provide an approach that can be integrated into test fill and test compaction tools to prevent

excessive power supply noise in delay tests and reduce noise-induced overkill.

In this dissertation, we propose a low-cost, vector-dependent modeling approach to analyze power supply noise and noise-induced delay. Two noise models are proposed to address circuits with array bond and wire bond package. This approach is then integrated into test fill and static compaction to control the supply noise level. In addition, we also propose a heuristic to estimate circuit transition count, which can be used to constrain noise during dynamic compaction.

Section 2 provides background and related work on various sources of noise. It particularly addresses power supply noise and lists all prior work in supply noise suppression, noise measurement and analysis. It also includes previous studies in delay variation with supply noise. Then, delay testing is briefly introduced including basic fault models and at-speed testing approaches. The last section lists a number of previous publications in delay testing addressing power supply noise, though they target different problems.

Section 3 introduces two power supply noise models for layout-aware noise analysis. Our noise models avoid complicated power network analysis, which significantly speeds up the supply noise analysis procedure. The two noise models are proposed to address array bond chips and wire bond chips, respectively. These two models are then compared in model application. Discussions on modeling off-chip current are also included.

Section 4 introduces algorithms for noise constrained static compaction based on the power supply noise analysis introduced in Section 3. It also includes a heuristic to

predict transition count based on partial assignment of input patterns, which can help control switching activity during dynamic compaction. Different test fill approaches are introduced, which are implemented in the experiments to show how delay varies with different test fill approaches.

Section 5 describes the experimental results. The experiments have been performed on both ISCAS benchmark circuits as well as an industrial design. First, all experimental data collected on ISCAS benchmark circuits is presented. It includes validation data of one noise model, noise constrained static compaction results that show how compacted test set size change with noise constraint, test fill data that show delay variation with test fill approaches, and transition count prediction results to address its efficiency in estimating switching activity. Then data on an industrial design is presented to show delay variation with different test fill and validation for the second noise model. The validation results of the two models are compared.

The last section gives conclusions and directions for future work.

2. BACKGROUND AND PRIOR WORK

2.1 Noise

Noise is inevitable in all electronic circuits. For decades, noise has always been of special interest to solid-state circuit designers and material scientists in the area of analog circuit design, where the noise generally comes from random motion of electrons in a resistive material, or the random recombination of holes and electrons in a semiconductor, or when holes and electrons diffuse through a potential barrier [4]. Typical types of noise considered in analog circuits are thermal noise, shot noise, flicker noise and burst noise [5].

In contrast to analog circuits, digital circuits are inherently immune to these noise problems through the use of high-gain logic gates, which restore logic values via nonlinear voltage transfer characteristics and significantly reduce the analog noise impact [6] [7]. However, the high gain of digital circuits has its own weakness and can result in much greater noise sources.

Noise in digital circuits first appeared as a problem in mixed digital-analog Integrated Circuits (IC) domain where noisy digital circuits strongly affect analog circuits [7] [8]. In the past decade, as technology scaled into the deep submicron (DSM) regime, digital circuits have also become more and more sensitive to noise, though the noise sources perceived in digital systems are quite different from the analog domain. As a consequence, noise analysis has become a critical concern for submicron digital circuit design.

For the purpose of noise analysis, noise can be roughly classified into two categories: functional and delay noise [9]. Functional noise causes performance degradation and signal integrity problems [10] [11]. If noise is of sufficient magnitude and can be propagated to a storage cell, such as a latch or a flip-flop, it can change the state of the circuit and cause functional failure. The goal of analysis of circuit behavior in the presence of functional noise is to verify that every signal line retains correct "1" or "0" values. Delay noise impacts the switching devices during signal transitions, thus changes the delay of the signal and affects the timing performance of the design [7] [12].

Noise can also be classified based on the noise source. There are various noise sources in digital systems [7]. The most relevant sources are:

- Leakage Noise
    - Subthreshold Leakage
    - Gate Leakage
    - Band-to-band Tunneling Leakage
- Charge-Sharing Noise
- Crosstalk Noise
- Power Supply Noise

There are several leakage mechanisms in the nanometer regime [13]. Three dominant components are: subthreshold leakage, gate tunneling leakage and reverse biased drain-substrate and source-substrate junction Band-To-Band-Tunneling (BTBT) leakage. Sub-threshold leakage is the leakage current from drain to source. It increases exponentially with the scaling of threshold voltage. Gate tunneling leakage is the leakage current due

to the tunneling of electrons or holes between the bulk silicon and the gate through the gate oxide potential barrier. As the oxide thickness scales to maintain reasonable Short Channel Effect (SCE) immunity, it also considerably increases direct tunneling current through the gate insulator of the transistor. In scaled devices, BTBT current through the reverse biased drain-substrate and source-substrate junctions also significantly contributes to leakage with higher substrate doping density and the use of "halo" doping profiles. However, the BTBT current can be reduced by SOI technology or other doping profiles. Leakage current has become a critical concern in power dissipation especially for low-power designs, and many leakage reduction techniques have been proposed [14] [15] [16].

The increased use of dynamic circuitry to achieve high speed and small area makes designs more vulnerable to noise problems [17]. Charge sharing is one of the problems that may cause failure in dynamic logic circuits due to their low noise immunity. Charge-sharing noise is produced by charge redistribution between internal nodes of the circuit. Techniques such as dual-rail logic and p-feedback/n-feedback transistors are often used to overcome charge sharing problems [18].

Capacitive coupling is the one of the primary noise sources in digital CMOS VLSI circuits. With technology scaling, signal lines that were once considered isolated can now interact with each other and significantly impact functionality and performance. One such interaction is known as capacitive crosstalk. It comes from parasitic coupling between adjacent signal nets and most likely affects the "victim" nets, the nets that have weaker drivers [19][20]. Capacitive crosstalk can either lead to logic failures, or

significantly increase propagation delay. Many approaches have been proposed to address capacitive coupling noise issues in design and test [21][22].

In additional to all of these noise sources introduced above, there is also power supply noise, which explicitly refers to the noise on the supply and ground network [7]. It is the focus of our work and will be explained in the following section.

2.2 Power Supply Noise

Power supply noise is the noise on the supply and ground network, which reduces the actual device voltage levels and increases signal arrival time at the primary outputs and next state lines [7][12].

Power supply noise consists of both DC and sinusoidal content. The DC noise, also termed IR drop, comes from resistive voltage drop due to wire resistance and the average static current demand of the chip [7]. In the case of DC noise, a DC network is built and solved to obtain the average IR drop at each location [23]. The sinusoidal noise, also termed as inductive ΔI noise, *di/dt* noise or simultaneous switching noise, comes from the RLC response of both chip and package due to switching current demands that peak at the beginning of the clock cycle [7]. The IR drop usually occurs on chip, while the inductive ΔI noise usually occurs on package. Therefore, these two different power supply noise sources are often treated separately, with different budgets.

In traditional analysis for power supply noise, the on-chip IR drop was the main focus, so most analysis tools modeled the on-chip power grid as a RC network. However, as we move into deep submicron design (DSM), the *di/dt* noise is becoming a

critical concern.

As operating frequency and gate density increase, simultaneous switching activity per unit area increases, which increases power density [3]. Meanwhile, DSM CMOS technologies require the use of reduced supply voltages [24]. Industry data shows that until recently, the power density of high-end microprocessors was increasing by approximately 80% per technology generation, with the voltage scaling by 0.8 [3]. This has led to higher current density, and consequently increased power supply noise. In the long run, *di/dt* noise will become dominant compared with IR drop, as it worsens with both increasing current demand and clock frequency [2]. Moreover, the requirement of cheap packaging, which means lower pin count and larger pin inductance, also causes larger *di/dt* noise [25]. Therefore, the *di/dt* noise is our main concern.

The *di/dt* noise may cause several types of errors in digital systems [25]. First and most important, *di/dt* noise will increase delay [26], especially on critical paths in pipelined circuits, and result in logic timing failures. Other problems includes phase-locked loop (PLL) jitter, which causes either timing errors due to clock skew or synchronization failures between different clock domains; I/O reference level problems; which may cause misinterpretation of input logic level or degrade circuit speed; and dynamic logic problems, since dynamic logic is quite sensitive to power supply noise.

2.2.1 Power Supply Noise Suppression

Many semiconductor companies, including AMD [27], IBM [28] and Intel [29], have included noise suppression techniques in their designs. Various techniques have been proposed to reduce power supply noise, *di/dt* noise in particular. Some of the techniques

makes changes to the circuits to generate less noise and improve noise immunity, while others suppress noise without modifying the circuits [25], requiring less time and fewer constraints in the design cycle. A most promising and widely used technique is adding on-chip decoupling capacitance between the power supply and ground.

The decoupling capacitance is usually made much larger than the capacitance of the switching devices, so it dominates power supply noise. A lot of research has been done on decoupling capacitor sizing and placement, either in the post-floorplanning stage or incorporated into floorplanning as a constrained maximum flow problem [30][31][32]. Recent work also proposed improved noise suppression techniques, such as active distributed decoupling capacitors [33] or active resistors in parallel with decoupling capacitors [34].

2.2.2 Power Supply Noise Measurement

Power supply noise measurement provides data on the noise occurring in chips. A possible measurement solution is to integrate power supply noise measurement systems on chip to characterize internal signals and noise behavior, which helps designers to improve and verify their designs [35]. These measurement circuits usually target special properties of supply noise, since it is very difficult to get a full time-domain voltage waveform during circuit operation. Muhtaroglu et al. proposed a circuit that targets overshoot and undershoot events [36]. Alton et al. presented a measurement technique to characterize the statistical properties and spectrum of power supply noise [35]. Another class of solutions integrates an online concurrent monitoring scheme to give warnings at the presence of excessive power supply noise, either for circuit diagnosis purposes or for

self-adaptation and correction [37][38]. Such schemes *s*hould be distributed across the whole circuit to observe power supply noise at any location and any given time.

2.2.3 Power Supply Noise Analysis

As the power supply noise problem becomes critical, the supply noise model is becoming one of the most interesting topics for researchers. An efficient supply noise model will help designers to gain a good knowledge of noise impact on circuit functionality and timing performance, and to improve the consideration of noise in design and test.

Power supply noise modeling and analysis generally is a challenging problem. Early research adopted a cell-based circuit model [39] and estimated power supply noise to calculate average power consumption, or investigated noise problem in a small circuit and scaled to larger designs [40]. However, a comprehensive package/on-chip power supply model usually consists of a number of circuit elements [41]:

- RLC model of package leads, ball grid arrays, power planes
- RC model of on-chip power interconnect
- RC model of intrinsic decoupling capacitance of non-switching devices and N-well regions
- RC model of explicitly designed decoupling capacitance
- Model of AC currents of switching devices

Much work has been proposed to cover the elements listed above. Chen et al. proposed a methodology to analyzed both resistive IR drop and inductive *di/dt* noise based on an integrated package/chip power bus model along with simulated switching

circuit model [11][42]. Power grid network analysis for noise estimation can be found in various other papers [43][44]. Nassif et al. introduced a novel Partial Differential Equation (PDE) related multigrid method for fast power grid simulation [45]. Zhu et al. also proposed a power network analysis method using a multigrid approach [46]. In 2003, Qian et al. proposed a fast and efficient power grid analyzer based on a random walk technique, which can be applied to both DC and transient analysis [47].

Despite these improvements, power network analysis remains an expensive approach, and it worsens with technology scaling. To save computation cost, some previous work adopted simple heuristics to estimate worst-case supply noise, such as switching transition count or sum of switching current [48][49]. These approaches can be used to simulate worst-case power supply noise when an accurate noise value is not required.

2.2.4 Delay Variation with Power Supply Noise

The voltage variations in the power supply network can have adverse impact on the timing performance of the circuit, and noise-aware timing analysis is a critical need. In order to develop a noise-aware timing analysis approach, we need to have a noise-sensitive cell delay model, and then integrate it in a comprehensive path delay model.

In general, the delay of a switching logic gate/cell is usually dependent on many factors, including supply voltage level, input voltage level, input slew rate, output capacitive load and other intrinsic design specifications. The sensitivity to supply noise increases as the supply voltage level declines. The rising delay is more sensitive to the voltage drop on the supply network while the falling delay to the ground bounce [48]. Early models usually adopted analytical approaches such that they represented gate

delay as an inverse or linear/quadratic function of the supply voltage [50]. Jiang et al. proposed a statistical approach to characterize delay [12]. Standard cell delay is treated as a perturbed random variable, and the probability distribution functions (PDF) are derived by simulating a set of characterization patterns. Look-up tables based on simulation are another widely used approach to model gate/cell delay [51]. Hashimoto et al. further studied the spatial power/ground level variation and proposed a power/ground (P/G) equalization method when the driver voltage level is different from the receiver, so that the gate delay model does not need to use input voltage levels as variables [52].

Path delay variation is roughly calculated as the summation of gate delay variations of all switching gates on the propagation path. Interconnect delay variation is not included since it is not sensitive to power supply noise. If input slew rate is also a variable in the gate delay model, the delay of each gate on the path is dependent on the output slew rate of its preceding gate.

A number of techniques have been proposed to compute the impact of power supply noise on timing performance. The static timing analysis (STA) technique in the presence of power supply and ground voltage variations was proposed to give the worst-case circuit delay [24]. Dynamic timing analysis, which predicts timing performance by simulating a set of selected patterns, is even more sensitive to power supply noise [53]. This is because power supply noise is highly dependent on the input vectors. In selecting the critical paths and generating patterns for dynamic analysis, one needs to consider the noise impact to ensure that the pattern set can produce realistic worst-case path delay.

2.3 Delay Testing

Timing performance is critical for high-end semiconductor products such as microprocessors. Testing is applied to integrated circuits after manufacturing to screen out defective parts from good ones. Most common defects are gross defects that produce incorrect values at primary output pins at any frequency, which is also called functional failure. However, some small manufacturing defects do not cause functional failures, but fail to produce correct values at the desired frequency. Delay testing is designed to detect such defects and ensure the parts that are shipped to customers meet the desired timing specifications [54].

Typical structural delay testing is performed as follows. Each delay test pattern contains two test vectors. The first vector initializes the circuit under test (CUT) and the second vector stimulates transitions on target signal lines, and makes sure a slow transition on these signal lines can be detected. A faulty circuit with a delay defect may pass a slow speed test but fail at higher frequency.

A fault model is an abstraction of a type of defect behavior. Classic delay fault models that are commonly used in delay testing are the transition fault model [55] and the path delay fault model [56]. The transition fault model assumes that the delay fault affects only one gate or line in the circuit under test, and can be detected on any sensitized path through the fault site. As the transition fault model targets relatively large delay faults, its test quality for small delay faults is a concern [57][58]. With the path delay fault model, a circuit is considered faulty if the delay of any of its paths exceeds the specified time. The path delay fault model is more realistic in modeling physical

delay defects, but the number of paths in the circuit can be exponential in the number of gates. Many techniques have been proposed to reduce the number of paths.

Test speed is another challenge in delay testing. Applying a test at the CUT functional speed is called at-speed test. Two scan-based at-speed test approaches have been widely used in industry. One is launch-on-shift (or skewed load [59][60]), which requires the second vector of the test pattern to be 1-bit shift of the first vector. The other is launch-on-capture (or functional justification, broadside [61]), which requires the second vector to be the system output of the first vector. The launch-on-shift approach has less test generation cost and higher coverage, but it requires a full-speed scan enable signal. On the other hand, the launch-on-capture approach does not require the scan enable signal to operate at full speed, and the sensitizable paths under the launch-on-capture constraints are also sensitizable in functional mode.

2.4 Power Supply Noise in Delay Testing

While noise-aware timing analysis has been thoroughly investigated in the past several years, the noise impact in delay testing has received only limited work. As more and more semiconductor manufacturers include at-speed delay testing into their product test flows, the noise impact on circuit timing, if not handled appropriately, may lead either to test escapes or test overkill. However, not much work has been published to address this issue.

Krstic et al. [62] [63] proposed a Genetic Algorithm based approach in pattern generation, which not only sensitizes and propagates the given fault, but also maximizes

power supply noise on path nodes and produces worst-case path delay. The results showed that their test patterns produced 19-59% extra path delay on average. However, the resulting maximum noise may be considerably greater than the functional mode worst-case noise. Moreover, this method set all don't care bits in the original patterns to a value, either "1" or "0", and this assignment of the don't care bits will very likely compete with other goals, such as crosstalk generation, that sometimes have greater impact on path delay or test power control.

Kokrady et al. [64] focused on timing validation for delay test vectors to avoid misclassification of good parts. Validation of test vectors is usually done by vector-based timing simulation, which invalidates and eliminates test vectors that cannot reliably distinguish between good parts and faulty ones. In their approach, the noise issue was taken into account during test validation, to improve the reliability of validated vectors. A layout-aware static method was proposed to validate at-speed transition fault delay test vectors in the presence of IR drop induced delay. However, inductive *di/dt* noise was not discussed in this work, although it usually dominates IR drop in current DSM designs.

Tirumurti et al. [3] proposed a fault modeling method that added power noise to a generalized fault model [65]. A vector-less approach was adopted to save expensive simulation time, so that actual simulation is performed on small cells to characterize peak switching current and current distribution on the power supply network, and the superposition rule is used for adding cell currents to estimate the impact of simultaneous switching activity. This work provides a comprehensive solution for switching cell

characterization, power grid analysis and fault identification, but the worst-case voltage drop is too pessimistic, and would never appear in functional mode.

3. POWER SUPPLY NOISE ANALYSIS

Section 2.2 introduced power supply noise, which is the noise on the supply and ground network that reduces the actual device voltage levels and increases signal arrival time at the primary outputs and next state lines. As we discussed in section 1, we need to design a vector-based modeling approach for power supply noise analysis that can be applied to restrain noise induced overkill in delay test.

The first and most important requirement for this modeling approach is short execution time. Power supply noise variation is largely dependent on the input pattern and the state of the circuit. To accurately characterize supply noise variation and noise-induced delay during delay testing, a vector-based approach is a must. However, during a vector-based approach, any slight computation cost increase per vector will be multiplied by the number of total vectors, and may eventually turn out to be a severe concern.

A lot of prior work has been published on power supply noise analysis to improve design and test while considering noise. Section 2.2.3 listed some of the approaches proposed in the past several years to characterize noise [11][39-47]. However, these approaches adopt vector-less strategies. Despite their comprehensiveness and accuracy, they are too expensive to be applied to vector-based analysis. Therefore, we propose a pattern-dependent solution for noise analysis that avoids heavy computation. The basic idea of our approach is explained as follows. We assume two-pattern delay tests. During the beginning of the launch clock cycle, when most switching activity occurs, the power pads are unable to provide current immediately to satisfy the switching current demand.

This is because off-chip inductance prevents the supply current from rising appreciably before most transitions have propagated. Therefore, during this period, most charge demanded by the switching devices comes from nearby, on-chip sources, such as parasitic capacitors and decoupling capacitors embedded in the circuit. The switching charge is finally provided by off-chip sources, but its impact on propagation delay is relatively small because most transitions complete before the off-chip current rises appreciably. This analysis ignores the background leakage current, since it is relatively constant.

With this basic idea, we conducted a series of experiments on ISCAS benchmark circuits and proposed our first noise model, which we identify as Noise Model I in the rest of the dissertation. It models on-chip activity as well as off-chip current based on an array bond power grid topology, so it works for array bond chips only. This model was found to be inadequate for peripheral bond chips in later research. We proposed a second noise model, which models on-chip activity more accurately, but neglects the off-chip current noise impact for simplicity. This second model was designed for peripheral bond chips, but can be extended to array bond chips. This model is referred to as Noise Model II in the rest of the dissertation. These two models are introduced and compared to each other in the following sections.

3.1 Noise Model I

Array bond chips, or area-array bond chips, adopts array bond approaches in the packaging process that distribute the chip I/O over the entire face of the die [66]. A

widely used array bond approach is Flip Chip. Flip Chip assembly is the direct electrical connection of face-down electronic components onto substrates, circuit boards, or carriers via solder bumps to the chip bond pads. It is first developed by IBM to assemble their mainframe computer modules in the 1960s, and the use of Flip Chip technology has grown rapidly in recent years. Array bonding is are highly effective in high performance systems. Its main disadvantage is high cost in manufacturing.

In contrast to array bond, peripheral bond chips require all the die I/Os to be in a single row or at most double around the periphery of the die. We will discuss peripheral bonding along with our Noise Model II.

3.1.1 Region

Power grid analysis [3] of array bond chips shows that the supply voltage impact of a switching transient is contained within a local area, since most current flows through nearby pads. Based on the topology of array bond chips, a region is defined as the area centered by a power pad, as shown in Figure 5. It is expected for an array bond design that each power pad should provide the current for the devices in its region. Hence, we start our first modeling approach based on this region concept.

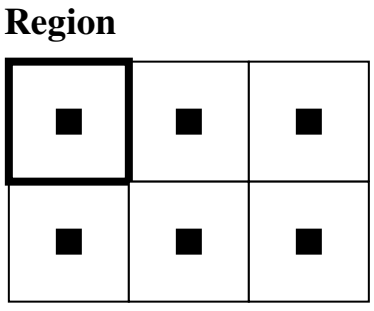

Figure 5. A region in an array bond chip.

3.1.2 Approximations

We make several approximations in our modeling approach [67][68. First, we assume that the supply voltage within a region is uniform, and the voltage of the regions is independent of each other. Further, we assume that the voltage drop for any cell in the region is identical. In addition, all switching activity across the region is equivalent, and any switching events outside the region can be neglected. The error of this approximation, along with several other approximations introduced later, will be estimated in experiments.

Our second approximation is that the on-chip switching current in a region, denoted as $I_{on\text{-}chip}$, comes from the on-chip decoupling and parasitic supply capacitance within the region. The decoupling capacitors are modeled as a single lumped capacitor between power and ground. The on-chip inductance is neglected for simplicity. On-chip wire resistance is also ignored in this model, so that the analysis becomes much easier than a traditional RLC network. Our model approximates the supply grid voltage as stepwise constant across the chip.

Third, we assume that the off-chip current in a region, denoted as $I_{off\text{-}chip}$, comes from a constant current source. This current source averages the previous K clock cycles of current consumption (based on the off-chip time constant). Thus, $I_{off\text{-}chip}$ must be taken into consideration if much switching activity occur in the previous cycles. However, during scan test, the scan-to-launch period is usually much longer than the functional mode cycle, so a chip is always in an idle state prior to the launch of a delay test vector. If the off-chip time constant is comparable to the scan clock cycle, $I_{off\text{-}chip}$ becomes

insignificant and can be safely ignored. More discussion on modeling off-chip current is included in section 3.4.

Fourth, voltage drop occurs on both supply and ground nets. A complete voltage drop analysis should take both networks into account. However, most prior work focuses only on the power supply network, with the assumption that power and ground can be separated [45]. We assume that the ground network is ideal, which means that ground bounce is not taken into account in this work. The techniques used here for the power network could also be used to the ground network to improve accuracy.

The leakage current is not considered here. This is because leakage current only affects static IR drop, which is almost the same from pattern to pattern. Therefore, it has no impact on our vector-based analysis and is not taken into account.

3.1.3 Noise Model I

Our simplified Power Supply Noise model within a region is illustrated in Figure 6. C_d is the distributed decoupling capacitance in a region, and C_p is the total parasitic capacitance of devices and interconnect within the region connected to the power supply network in the current clock cycle. All switching cells that draw current from the supply within this region during the clock cycle are modeled as time-varying current sources $I_{switching\_i}$. The switching current model is discussed below. $I_{on\text{-}chip}$ is the switching current provided by the on-chip capacitance, and $I_{off\text{-}chip}$ is the switching current provided by the power pads.

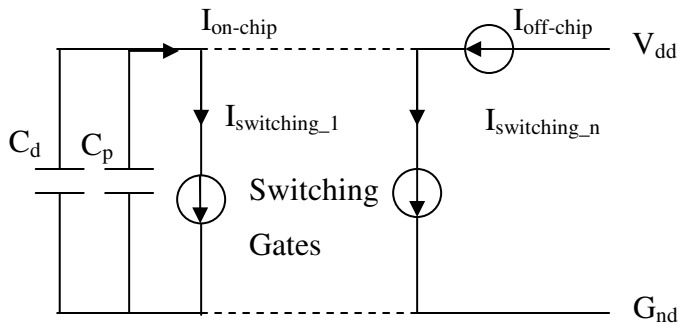

Figure 6. Supply noise model within a region (Noise Model I).

The maximum voltage drop in this region during a clock cycle, ΔV_{max}, is:

$$\Delta V_{max} = (\int I_{on\text{-}chip}) / (C_d + C_p) \quad (1)$$

$$\Delta V_{max} = ((\sum \int I_{switching\_i}) - \int I_{off\text{-}chip}) / (C_d + C_p) \quad (2)$$

After most switching transitions occur, voltage recovers through $I_{off\text{-}chip}$ until the start of the next cycle.

The flow chart of the noise analysis procedure is illustrated in Figure 7. To estimate the power supply noise effect of a delay test vector (a vector pair for delay faults), we first use logic simulation to find transitions on all nets in the circuit. Layout information is then needed to estimate voltage drop for each region. In practice, only those regions traversed by the targeted path need to be considered. We then calculate path delay with our delay model.

The time complexity for this procedure is *O(cell_count)*, where *cell_count* is the total number of cells of the circuit. This means that our analysis approach has the same time complexity as logic simulation.

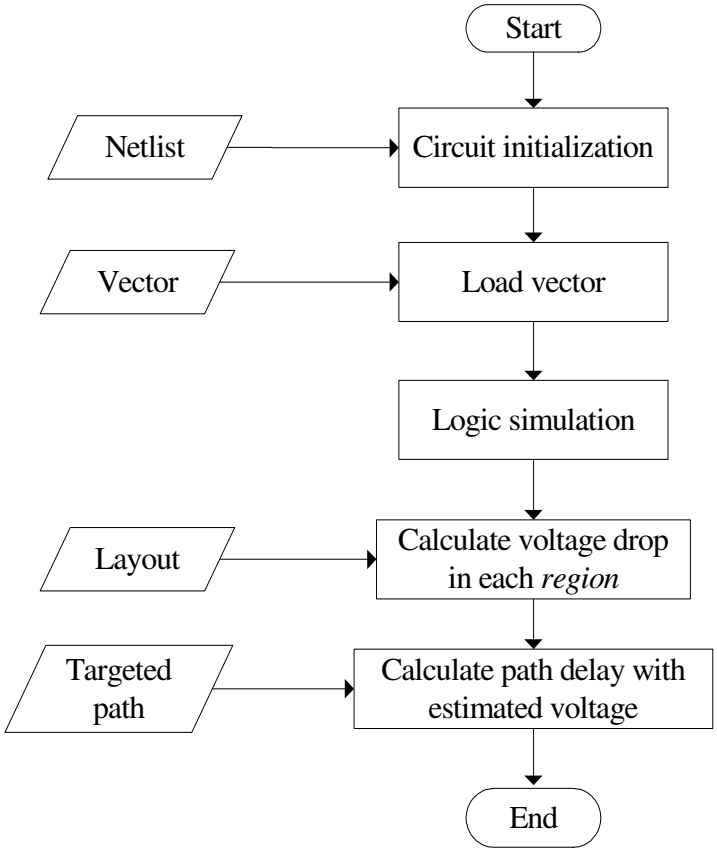

Figure 7. Power noise analysis procedure.

3.2 Noise Model II

Although area-array bonding technology prevails in high-performance chips, its high manufacturing cost prevents it from replacing peripheral bonding approaches in cost-driven applications. Peripheral bonding approaches, most often wire bonding, require all the die I/Os to be in a single row or at most double rows around the periphery of the die

[66]. It is an older technology that uses face-up chips with a wire connection to each pad. Today, wire bonding is still considered the most cost-effective technology in assembly and packaging, and is used for the vast majority of semiconductor products.

Noise Model I cannot be applied to peripheral bond chips. The definition of a region, on which Noise Model I was based, was the area centered on each power pad. Obviously, for peripheral designs, this definition does not work. Therefore, a model that more accurately characterizes localized voltage variation is necessary.

3.2.1 Effective Region

Here we propose a new concept for Noise Model II [69. The circuit is first extracted as a large RC network. On-chip inductance is neglected since it is relatively small compared to the package inductance. Assume a current impulse occurs somewhere in the network. Capacitors around this impulse will begin to discharge in order from nearby to far away, and result in localized voltage drop. However, if a capacitor is far enough away, it is possible that it will not discharge within the clock cycle. Such capacitors should be considered irrelevant to the noise analysis. Consequently, an effective region for a switching device is defined as the area whose RC time constant is less than or equal to the clock cycle time. Put another way, a capacitor only provides current to devices whose effective regions cover that capacitor.

The RC time constant T of a region follows from the integration over the region area of the supply network resistance times the circuit capacitance, which is recently presented by Paul van de Wiel et al.[70]. To introduce the calculation for RC time constant, we first start with an annular metal board in Figure 8, and come up with the

following equations:

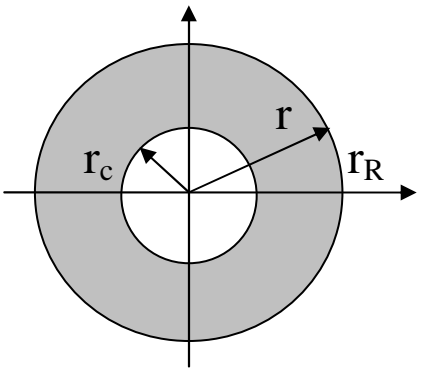

Figure 8. An anualar-shaped metal board.

$$dR = \rho \cdot dr \ / (2\pi r) \quad (3)$$

$$dC = c \cdot (2\pi r) \cdot dr \quad (4\})$$

where ρ is the sheet resistance, c is the average capacitance per unit area, r is the distance from the center, and R and C are resistance and capacitance as a function of r, respectively [70]. Based on equations above, the following function can be derived to compute RC time constant T as a function of r:

$$T = \int d(RC) = \int (R \cdot dC + C \cdot dR) = 0.5 \cdot \rho \cdot c \cdot \ln(r_R / r_c) \cdot (r_R^2 - r_c^2) \quad (5)$$

where r_c is the inner radius and r_R is the outer radius of the annular sheet. These equations can also be extended to the circuit, which is not annular-shaped or evenly distributed. Approximations are necessary in this extension. The power grid can be approximated as a metal sheet by using the metal fill rate when computing sheet

resistance. The inner radius r_c is set to half of the diameter of a first-level contact, which is usually much smaller than the radius r_R of the area, so the function can be approximated as:

$$T \approx 0.5 \cdot \rho' \cdot C_A \cdot \ln(r_R / r_c) \quad (6)$$

where ρ' can be computed by dividing the metal sheet resistance by metal fill rate and number of layers, and C_A is the total capacitance of the whole area. However, the parasitic capacitance included in C_A is pattern-dependent and makes the RC time constant and so the effective region pattern-dependent as well. In order to compute a pattern-independent effective region, C_A is approximated as the total decoupling capacitance of the area times a ratio. The ratio is defined as the whole chip signal net capacitance plus decoupling capacitance, divided by the decoupling capacitance. The assumption is that the ratio of signal net and decoupling capacitance is similar in each region.

Most of time, the majority of switching activity is completed within the first half clock cycle. Therefore, it is also valid to use half of the clock cycle time in deciding the size of the effective region.

An algorithm has been developed to define effective regions for all devices on the chip. This algorithm only needs to be applied once for each circuit. Its flowchart is shown in Figure 9.

If two switching devices, or two decoupling capacitors, are very close to each other, it is obvious that they will have very similar effective regions and voltage variation. These devices can be put together for analysis to reduce computation complexity. Therefore,

we first divide the whole chip area into m × n small squares or grids, each containing a limited number of capacitors and switching devices. These grids will then be assigned to effective regions, which will determine the effective region for all devices and capacitances within the grid. The grid size is chosen such that the effective region can be accurately determined. In practice, the grid size can be quite large as long as its RC time constant is small compared to the clock cycle time.

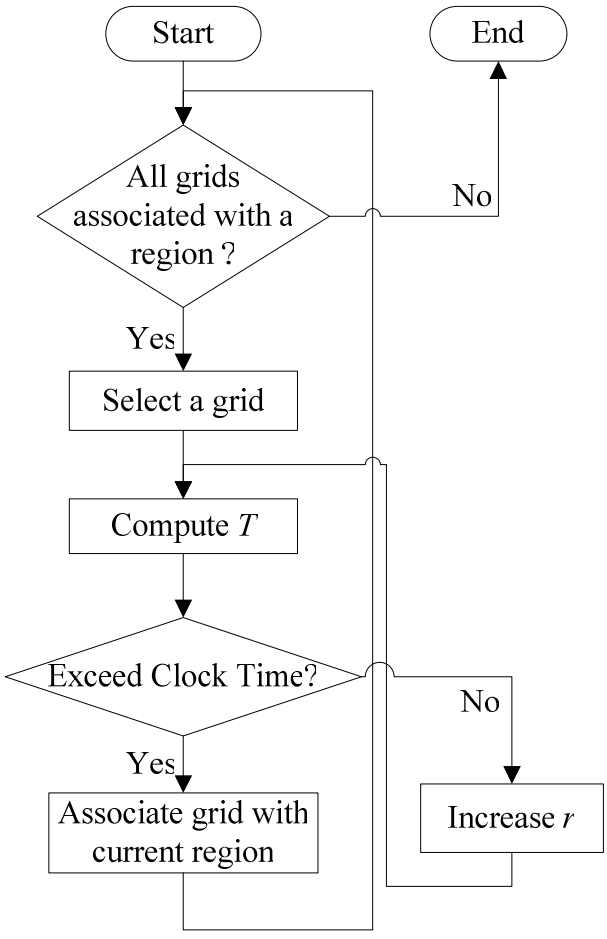

Figure 9. Flowchart of the algorithm to find effective regions for all devices.

To determine the region associated with each grid, we start with the grid itself, and then increase the radius by one grid width each time to expand the region until the RC time constant equals or exceeds the clock cycle time. Some grids are only partially covered, but they are still considered part of the region as long as over 50% of the grid area is covered. We repeat this analysis for all grids until each has an effective region.

The complexity of the effective region algorithm is $O(grid\_count^2)$, where *grid_count* is the total number of grids As discussed above, grids must have RC time constants small compared to the clock cycle time to achieve good accuracy. In our experiments, we have found that we can achieve this accuracy by setting *grid_count* to the square root of cell count, so that the complexity of the algorithm is $O(cell\_count)$.

3.2.2 Approximations

As with Noise Model I, we make several approximations before introducing our noise model.

First, the voltage level (and power supply noise) is uniform within each grid. Therefore, the voltage level for all cells in the grid is identical. This approximation is reasonable, since the spatial voltage variation within a small area is small, due to embedded capacitance and low resistance. This is different from our approximation for Noise Model I, which assumes uniform voltage in the whole region.

Second, in response to a switching impulse, all capacitors in the effective region are assumed to be equally effective, despite their varying distance to the switching device. Therefore, the total switching charge in the grid is evenly provided by all capacitors in the effective region. For each grid in the region, the percentage of total charge it needs to

provide for the center grid depends on the ratio of its capacitance to that of the whole region. Further, parasitic capacitance is approximated as constant, since. experiments show that the pattern-to-pattern variation of parasitic capacitance is small. This approximation makes the effective regions independent of test patterns.

A third approximation is that there is no current coming from off-chip sources. As we discussed before, the power supply cannot response immediately to the impulsive switching current demand, due to high package inductance and the long idling time during the scan cycle prior to the launch cycle. Approximately, most switching activity occurs in the first half of the clock cycle. For example for the chip in [71], the average path length is 3 ns, while the longest path is 7 ns. Therefore, the off-chip current is considered insignificant compared to on-chip current demand when most transitions are propagated. However, this approximation can be replaced by more accurate off-chip current modeling approaches, which are discussed in section 3.4.

3.2.3 Noise Model II

Our simplified noise model within a grid is illustrated in Figure 10. As we have discussed, each grid contains two kinds of components: capacitors and switching devices. A grid provides current by discharging its capacitance for any switching devices whose effective region covers this grid. In the meantime, it absorbs current from all capacitors in its effective region. Similar to Noise Model I, C_d is the distributed decoupling capacitance in a grid, and C_p is the total parasitic capacitance of devices and interconnect connected to the power supply network in the current clock cycle. All switching cells that draw current from the supply within this region during the clock

cycle are modeled as time-varying current sources, which will be discussed in Section 3.5.

The maximum voltage drop for a particular grid during a clock cycle is:

$$\Delta V_{max} = (\sum(\alpha_i \cdot Q_i)) / (C_d + C_p) \quad (7)$$

where Q_i is the total switching charge of grid i, whose effective region covers the current grid, and α_i is the ratio of the decoupling capacitance of the current grid to the whole effective region of grid i.

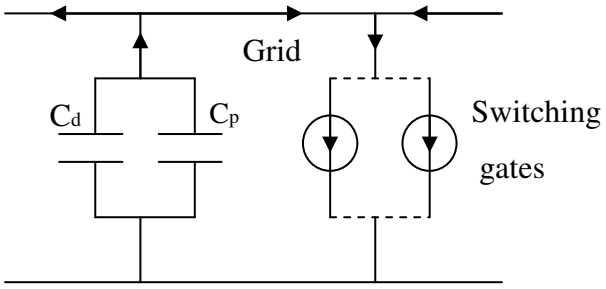

Figure 10. Supply noise model within a grid (Noise Model II).

Figure 11 is the flow chart of the entire noise analysis procedure for one test pattern. We first load the circuit netlist and layout to locate devices and extract parasitic capacitance. Each grid is then associated with an effective region. This initialization only needs to be performed once per circuit. Then for each test pattern, logic simulation is applied to find transitions on all signal nets of the circuit. We use zero-delay simulation in either noise model, but we can also apply a timed logic simulation to accurately estimate glitches using a back-annotated Standard Delay Format (SDF) file. Note that a

glitch is also considered as a full rising and a full falling transition in our work. We then calculate the switching charge needed for each grid, and distribute the charge among all the grids in its effective region. Equation 3 introduced above can then be applied to each grid to calculate power supply noise. Delay models are applied in the last step of the analysis to calculate path delay with noise impact.

The time complexity for this procedure is $O(cell\_count + grid\_count^2)$. As discussed in section 3.2.1, *grid_count* can be of the order of the square root of *cell_count*. In addition, finding effective regions for grids only needs to be performed once per circuit. Hence the complexity becomes $O(cell\_count)$. This means that our noise estimation approach has the same time complexity as logic simulation.

3.3 Model Comparison in Model Application

For each test pattern, a complete power-noise-aware timing analysis can be classified into two consecutive steps: 1) compute the on-chip voltage level, and 2) compute the propagation delay on the critical paths. The analysis flows for the two noise models both follow this scheme, yet there is still some difference in model application. The analysis flows of the two models are compared in Figure 12.

The time complexity of power supply noise analysis per test pattern is O(*cell_count*) for Noise Model I, where *cell_count* is the total number of cells of the circuit. For Noise Model II, the time complexity is $O(cell\_count + grid\_count^2)$. In practice, *grid_count* can be of the order of the square root of *cell_count* with slight impact on accuracy. Hence the complexity also becomes $O(cell\_count)$, the same as Noise Model I.

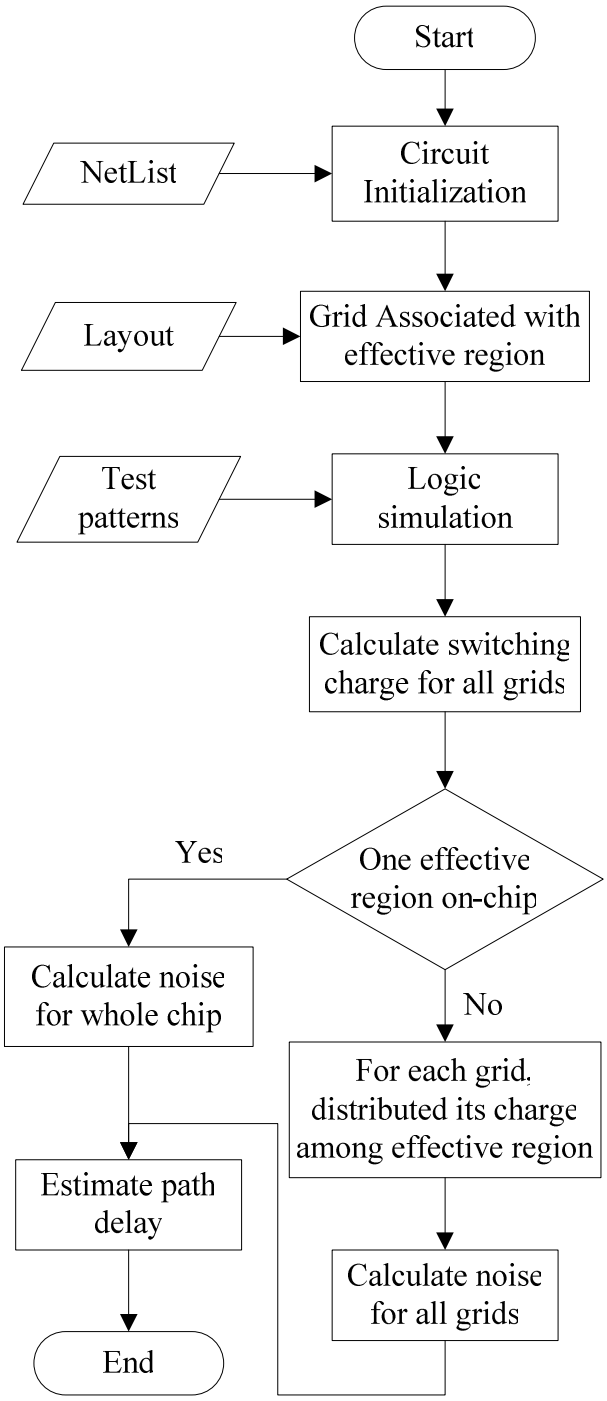

Figure 11. Flow chart for power supply noise analysis.

3.4 Discussion on Off-chip Current Modeling

Because Noise Model II better characterizes local voltage variation, it has the potential to be more accurate. However, a major drawback in Noise Model II is that it does not consider off-chip current. Off-chip current has always been a difficult problem in either of our noise models. In general, three approaches can be applied to address off-chip current.

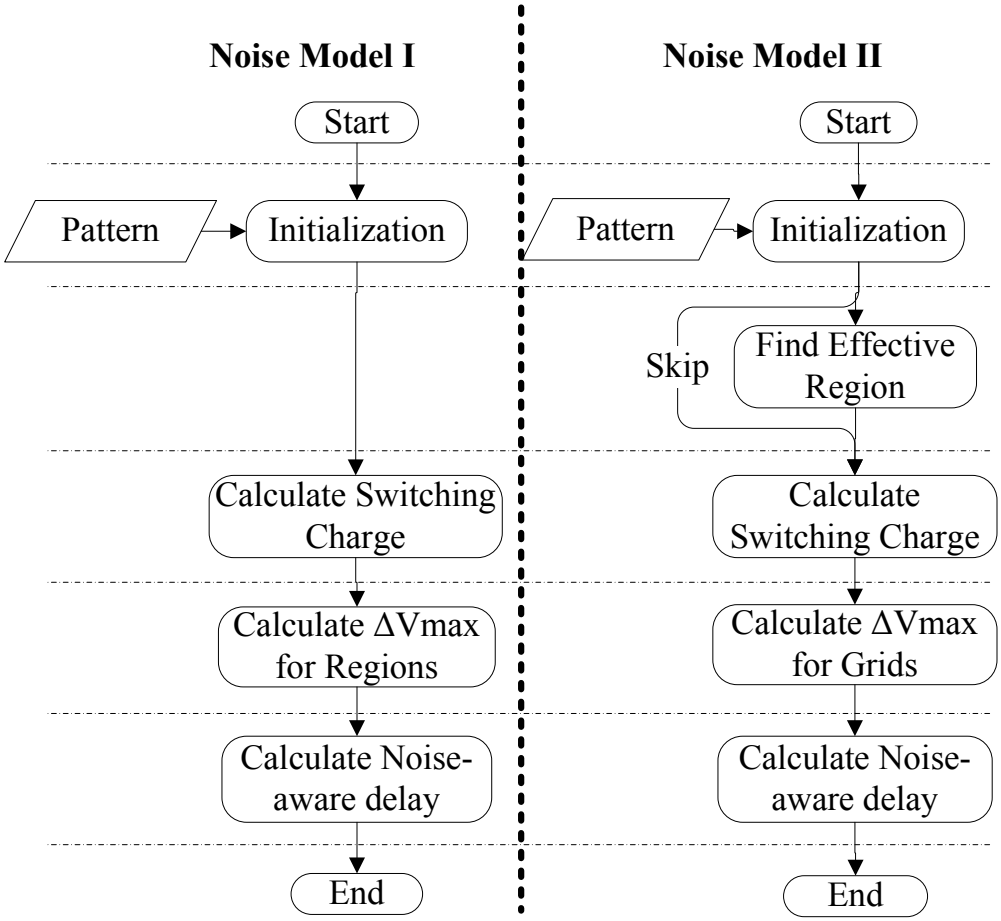

Figure 12. Analysis flow comparison for Noise Models I and II.

First, off-chip current can be neglected. We adopt this approach in Noise Model II for simplicity. As mentioned before, the power supply cannot response immediately to the impulsive switching current demand, due to high package inductance and the long idling time during the scan cycle prior to the launch cycle. Also for peripheral chips, most switching devices are not close to the power pads, which limits the impact of off-chip current. This strategy relieves us from analyzing the off-chip current effect in reducing voltage drop. However, if non-scan testing is used, or if the idle time before the launch cycle is short enough, this approach will no longer be accurate.

Second, off-chip current can be modeled as a constant current source, since the off-chip time constant of the power supply is much larger than the system clock cycle due to high package inductance. This current source averages the previous K clock cycles of current consumption (K is based on the off-chip time constant). Noise Model I adopts this approach. For scan test, this approach will give us a close-to-zero constant off-chip current due to the long scan cycle before the system clock cycle, which is similar to the first approach. This approach takes previous circuit state into account, hence it is a more accurate model of off-chip circuits. The disadvantage is that it does not consider the impact on off-chip current from switching activity in the current clock cycle.

Third, we can model the off-chip current with a theoretical approach. Three assumptions are made to validate this approach:

- All switching activity is finished by the first half of the clock cycle. No switching activity occur in the second half of the clock cycle.
- Switching charge demand increases linearly during the first half of the clock

cycle.

- On-chip voltage level is uniform across the chip.

Therefore, we have the following equations based on charge conservation, assuming the initial off-chip current is 0 (which is valid for scan-based test with long scan cycle prior to the launch cycle):

$$\begin{cases} CL\left.\frac{di_1}{dt}\right|_{t_x} + \int_0^{t_x} i_1 dt = \alpha \cdot t_x \\ i_1(t=0) = 0 \\ t_x < t_c / 2 \\ \alpha = \frac{2Q_s}{t_c} \end{cases} \tag{8}$$

in which i_l is the off-chip current from the power supply, C is the on-chip capacitance, t_x stands for a time point in the first half of the clock cycle, t_c is the system clock cycle time, Q_s is the total switching charge demand from the circuit, and α is the linear factor of switching charge demand increase.

And we get:

$$i_1 = \alpha(1 - \cos(t/\sqrt{CL})) \tag{9}$$

$$\Delta v(t) = \alpha\sqrt{\frac{L}{C}}\sin(t/\sqrt{CL}) \tag{10}$$

In case the initial off-chip current is non-zero, we have:

$$\begin{cases} CL\left.\frac{di_1}{dt}\right|_{t_x} + \int_0^{t_x} i_1 dt = \alpha \cdot t_x \\ i_1(t=0) = i_{init} \\ t_x < t_c / 2 \\ \alpha = \frac{2Q_s}{t_c} \end{cases} \tag{11}$$

The results become:

$$i_1 = (\alpha - i_{init})(1 - \cos(t/\sqrt{CL})) + i_{init} \tag{12}$$

$$\Delta v(t) = (\alpha - i_{init})\sqrt{\frac{L}{C}}\sin(t/\sqrt{CL}) \tag{13}$$

In which i_{init} is the initial value of off-chip current when the launch clock cycle is applied. It can be the current that averages the previous K cycles of current consumption, as we did in the second approach. In this way, we combine previous circuit state impact with current circuit state impact in analyzing off-chip current, and this approach should be more accurate than the previous two.

These three approaches are all applicable for Noise Model I. However, the second and the third approaches, which are regarded as more accurate, can not be directly integrated into Noise Model II. What is missing from these two approaches is how the off-chip current is distributed on-chip, and how many switching devices and how much area it affects in reducing voltage drop.

A practical method is to view the power pad as a current source similar to any on-chip switching devices, but negative in value, and find an effective region for it. In this way,

we can perform the same analysis for off-chip current as for on-chip switching devices. The second approach, which averages previous cycles of current consumption, can be integrated to Noise Model II in this way. The third approach, which should be the most accurate, needs further modification if it will be adopted in Noise Model II in future work.

3.5 Switching Models

We must calculate $\int I_{switching\_i}$ for each library cell, or Q_i for each grid in Noise Model II, in order to compute worst-case voltage drop. Switching current drawn from the supply network in CMOS circuits consists mainly of two parts, the short circuit current and the charging/discharging current on the output capacitive load. The latter term is usually the dominant term, due to slew rate design constraints.

3.5.1 Dynamic Charging/discharging Current

Charging/discharging current in CMOS circuits is well understood and easy to estimate. Tirumurti et al. [3] created a table of peak power and ground currents for different values of cell output load and input slope by simulation. This approach incorporates both short-circuit and charging current. We adopt a similar approach to calculation charge due to dynamic charging current. Figure 13 shows a typical waveform for an inverter. This waveform is usually approximated as triangular in order to compute the total charge of each transition, as shown in Figure 14. If the load is quite large, sometimes this waveform is also approximated as a trapezoid.

Based on the triangular approximation in Figure 14, a table is built by simulation for each cell, such that one can determine its peak current and output transition time for different values of output load and input slew rate. Once we get the peak current and transition time from the table, the total charge demanded by a transition can be calculated as:

$$Q = 0.5 \cdot I_{peak} \cdot (t_{end} - t_{begin}) \quad (14)$$

where I_{peak}, t_{end} and t_{begin} are computed from simulation. The input slope during circuit operation is unknown, since we do not know the actual input slope for each gate before estimating voltage drop and apply our delay models. Instead, we can use the input slew rate from static timing analysis, assuming nominal delay.

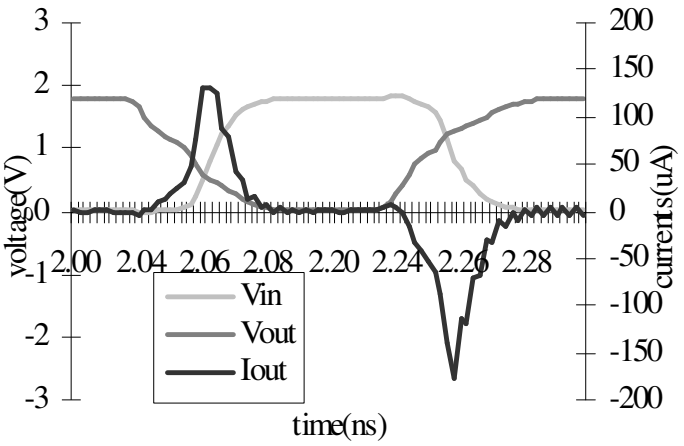

Figure 13. Charging/discharging current waveform for an inverter.

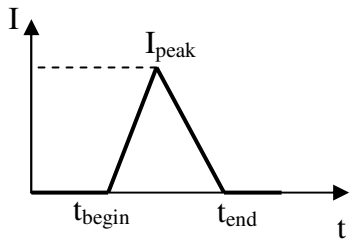

Figure 14. Switching current model of dynamic charging current for CMOS devices.

3.5.2 Short Circuit Current

During switching in a static CMOS logic gate, a direct path from the power supply to ground is established [72] that results in short circuit current. Short circuit current is dependent on the input rise/fall time, the load capacitance and gate design. When the load capacitance is small enough, the short circuit current dominates the current drawn from the supply network. Similar to charging/discharging current, we can also create a table of peak current for different values of gate output load and input slope by circuit simulation. Similarly, the input slope for each gate is computed by static timing analysis assuming nominal delay. The current waveform is approximated as triangular, which is accurate in most circuit designs, particularly low power designs.

Another practical approach is using analytical functions to calculate short circuit charge for all cells, since most of the time short circuit current is relatively small. The short circuit current model used here is based on previous work by Sylvester et al [73][74]. By making various assumptions and approximations, the peak current is

substituted with a certain fraction of the saturation current, and the time of short circuit current is approximated as:

$$T_{short} = 1.1 \cdot (R_d \cdot (C_j + C_{in}) + R_d \cdot C_w + R_w \cdot C_{in} + 0.4 \cdot R_w \cdot C_w) \quad (15)$$

where T_{short} is the flow time, R_d and R_w are the device and wiring resistance, and C_j, C_{in} and C_w are the junction, input and wiring capacitance, respectively. Assuming a triangular waveform for the short circuit current, we can calculate the short circuit charge using both T_{short} and peak current.

3.6 Delay Models

Power-noise-aware timing analysis consists of two consecutive steps: computing the on-chip voltage levels and computing the propagation delay on target paths considering noise impact. The first step was included in the two noise models. Here we focus on propagation delay computation with noise.

Several delay definitions must first be given. Cell delay is measured as the time interval between the input crossing approximately $V_{dd1}/2$ and the output crossing approximately $V_{dd2}/2$, where V_{DD1} and V_{DD2} are the input and output voltage ranges of the cell. For both input and output, the accurate measurement point is the 40% point for rising transitions and the 60% point for falling transitions. The transition time is specified in the 10% to 90% interval of full swing. Some prior work also suggests the 30% to 70% interval is more accurate [75].

Several models have been proposed for delay functions. Bai et al. proposed the following delay equation [76]:

$$t_d = A + BV_{DD} + CV_{DD}^2 \quad (16)$$

where the coefficients can be obtained from simulation. The coefficients here strongly depend on the input transition time and output capacitance. Bai et al. also suggested linear functions of supply voltage with appropriate coefficients if the voltage drop is not too large [76].

Another widely used delay modeling approach is to model both delay and transition time as a function of input slope, output capacitive load and device voltage level. This approach was applied in our experiments for Noise Model I. The models are generalized as follows.

$$t_d = f(t_{in}, C_{out}, V_{dd}) \quad (17)$$

$$t_{out} = g(t_{in}, C_{out}, V_{dd}) \quad (18)$$

where t_d is the gate delay, C_{out} is the output load, t_{in} is the input transition time and t_{out} is the output transition time.. A table method based on these equations has been used to calculate t_d and t_{out}.

A third delay model, which we used in the experiments for Noise Model II, models both delay and transition time as a function of input slope and output capacitive load first. A look-up table is built in this step. Then simulations are performed for each library cell to find a linear relationship between delay or slope and voltage level. Rising and falling transitions on the output are considered separately. We then use the linear model to calculate real delay and slew rate:

$$t_d = \beta \cdot f(t_{in}, C_{out}) + \gamma \quad (19)$$

$$t_{out} = \lambda \cdot g(t_{in}, C_{out}) + \zeta \quad (20)$$

where t_d is the cell delay, t_{in} is the input transition time and t_{out} is the output transition time. β, γ, λ, and ζ are coefficients from simulation. They will have different values for rising and falling transitions.

Different operating conditions may have a significant impact on delay. The main factors in operation conditions are nominal voltage, temperature and process parameters. Different delay models may be necessary when operating conditions change.

The supply voltage varies during a clock cycle due to supply noise. The voltage level during the logic transition can be regarded as constant, if the time constant of the noise waveform is much larger than the transition time [52]. However, it is difficult to know the actual voltage level on a device during its transition, unless we know the real noise waveform and the real time of the transition. Therefore we assume that the supply voltage level drops linearly with time during the clock cycle, and the worst case voltage drop occurs when all switching activity finishes. This assumption is based on an approximation that most paths are of similar length, so that the switching density is uniform until the time that most switching activity finishes. At that point, there are only a few long paths still propagating, as in [71]. We further use the voltage at the nominal switching time of the device, since we do not know the actual switching time.

The supply voltage varies both temporally and spatially. In real designs, gates in a path are not necessarily placed in the same neighborhood. If two gates, one a driver and

the other a receiver, are placed far from each other, their supply voltage levels are very likely to be different [52]. A different input voltage level, other than supply voltage, may affect the charging/discharging current and eventually affect delay.

Since there are multiple inputs in many gates, the cost of characterization by simulation is prohibitive. An equalization method to model different driver and receiver supply voltages was proposed by Hashimoto et al. [52]. Since gate delay is the time to charge/discharge the gate output load and voltage level variation causes gate delay variation by changing the charging/discharging current, gate delay can be kept unchanged by increasing/decreasing the output load in the same ratio. DC analysis was performed varying all input voltage levels and a Response Surface [77] was built for charging/discharging current before and after voltage level equalization. The current ratio is then used to compute the replaced output load value. Since voltage levels of all inputs have already been equalized, only the device voltage level, output load and input slope will be taken as variables for gate delay calculation.

Spatial voltage variation is not taken into account in our work due to unpromising experimental results. We simply assume that the device input voltage level is the same as the device voltage. That is, we do not consider the delay effects of different driver and receiver supply voltages.

4. COMPACTION AND FILLING STRATEGIES

4.1 Basics of Compaction

Compaction is a technique to simultaneously apply a set of test patterns with non-conflicting values to reduce test set size for combinational circuits, or test sequence length for sequential circuits [78]. Reduced test set size or test sequence length results in less test application time. This is especially crucial for scan-based designs, since the test application time for these circuits directly depends on test set size and scan chain length [79]. Besides, it also reduces tester memory requirement, which is one of the main factors of test equipment cost [80].

To detect targeted faults, automatic test pattern generation (ATPG) tool only needs to specify values on a subset of all primary inputs (plus all scan flip-flop outputs for scan designs), and leaves the rest of the values as don't care. Those don't care bits can be set to either "0" or "1", as needed to enable compaction.

The compaction operations for two one-bit vectors are shown in Table 1 [78]. X stands for don't care, and Φ stands for invalid compaction. If the vectors are both 0 or both 1, or at least one of the vectors is don't care, these two one-bit vectors are compatible and can be compacted. Two test patterns are compatible if and only if they are compatible for every bit.

Compaction algorithms based on several heuristics have been proposed for both combinational circuits [79][81][82][83][84] and sequential circuits [78][80][85][86][87][88][89]. Most of the techniques focus on the stuck-at fault model.

Several papers address compaction of two-pattern tests for transition fault, stuck open fault and path delay fault [79][90]. Sankaralingam et al. discusses compaction techniques to control scan power dissipation [91]. However, none of them address noise issues.

Table 1. Compaction operation of two bit vectors.

	0	1	X
0	0	Φ	0
1	Φ	1	1
X	0	1	X

Two categories of compaction techniques exist: static compaction and dynamic compaction. Static compaction techniques are applied after test generation, while dynamic compaction techniques are applied concurrently with test generation. The following two sections address noise consideration in static and dynamic compaction processes, respectively.

4.2 Noise Consideration in Static Compaction

Static compaction seeks to reduce the test set size after it has been generated. It is a post-processing step to test generation [80][88]. It is independent of the test generation process, so it does not require any modifications to the ATPG tool. Moreover, static compaction can be applied to further reduce the test set size even if dynamic compaction was used during test generation. Static compaction is an effective technique in reducing

test set size and also is easy to apply. Various static compaction algorithms have been proposed [78][80][85][88][89][91][92].

The key goal in our compaction work, different from all previous work, is that the power supply noise effect for all compacted delay test patterns should be within the functional mode level, with compaction rate only the second concern.

An important property of uncompacted delay test patterns, path delay test patterns in particular, is the low care bit density. Qiu et. al. presented care bit density data of transition fault test and path delay test (after compaction) on an industrial design [93]. Even after compaction, the average care bit density for transition fault test was 4.59%, while for path delay test, it was as low as 2.23%. Obviously, the care bit density for uncompacted transition fault test patterns and path delay test patterns is even lower. This low care bit density brings freedom in modifying compaction to reduce power supply noise, while it also limits the negative impact on compaction rate.

Therefore, we want to build a static compaction framework, and then integrate noise constraints into this compaction procedure to generate noise limited compacted tests. In the following two sections, section 4.2.1 explains the basic static compaction algorithm without noise concerns, and section 4.2.2 introduces how noise is constrained during compaction.

4.2.1 Static Compaction Framework

In our work, a simple greedy static compaction strategy is used. Test patterns are considered one by one in order and combined with the first compatible pattern found in the compacted pattern list, as shown in Figure 15. This is called the forward order greedy

algorithm in the rest of the section. Backward order greedy algorithm is similar except the test patterns are combined with the last compatible pattern in the compacted pattern list. We also implemented a static compaction tool using simulated annealing in order to find a close-to-optimal solution for compaction. Our experiments show that the results of greedy algorithms are close to optimal while taking much less time than simulated annealing.

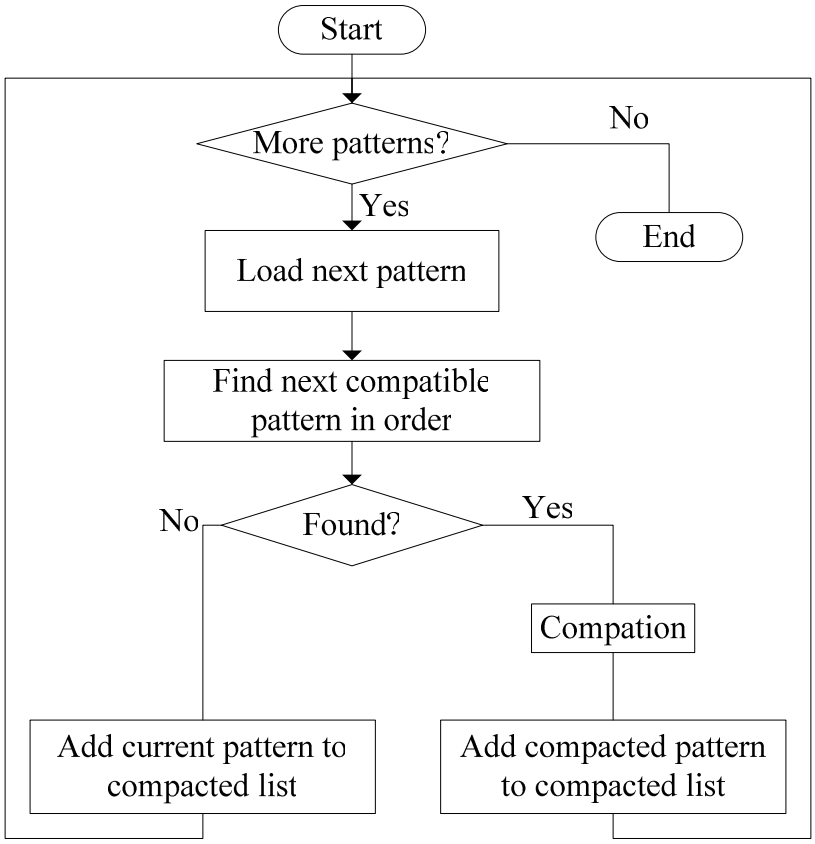

Figure 15. Flow chart of compaction using greedy algorithm (w/o noise constraints).

Experiments were performed on several ISCAS89 benchmarks and an industrial circuit, “Controller 1”. The test patterns, generated by the CodGen, a path delay fault

ATPG tool [94][95], were launch-on-shift robust path delay tests targeting the longest rising and falling transition path through every line in the circuit (termed KLPG-1).

The comparison results are shown in Table 2. Column 1 lists the benchmarks, and column 2 shows the initial test set size from CodGen. Two greedy algorithms are implemented. Greedy I loads vectors for compaction in a forward order, while Greedy II uses a backward order. Columns 3 and 4 list the compacted test set size and compaction time, respectively, of the Greedy I algorithm, and columns 5 and 6 show results of the Greedy II algorithm. The smaller test set is selected from the two and shown in column 7. The simulated annealing algorithm, denoted as SA, chooses two vectors at random for compaction and uses a weighted heuristic to determine whether the move is accepted. Similarly, the compacted test set size and running time of Simulated Annealing are shown in columns 8 and 9, respectively. Column 10, the last column, lists test set size increase using greedy algorithm vs. simulated annealing data in columns 7 and 8.

Table 2. Compaction results for greedy algorithms and simulated annealing.

Circuit	Initial Test Size	Greedy I forward		Greedy II backward		Greedy Test Size	Simulated Annealing (SA)		Greedy vs. SA
		Test Size	Time (s)	Test Size	Time (s)		Test Size	Time (s)	
s1423	395	216	3	215	4	215	212	300	1.4
s1488	192	88	1	86	1	86	85	1,457	1.2
s1494	193	86	1	84	1	84	83	1,504	1.2
s13207	3,220	916	46	899	82	899	901	6d	0.2
Contro-ller 1	12,274	2,325	405	2,232	892	2,232	2,203	30d	1.3

The greedy approach generates 1-2% more tests than simulated annealing. The efficiency of the forward greedy algorithm is almost the same as the backward algorithm. It generates about 100 more patterns than the backward-order algorithm for controller 1, but costs less than half the running time. Therefore, the forward-order greedy algorithm was chosen for compaction framework implementation.

4.2.2 Noise Constrained Static Compaction

As mentioned above, the key goal of our compaction tool is to guarantee that the power noise effect for all compacted test patterns is within the functional mode level, with compaction rate only the second concern. There are various ways to define the functional mode noise level. The simplest approach is to use the maximum voltage drop specified by the power grid designer. If silicon is available, an empirical approach is to apply functional patterns to the circuit using automatic test equipment (ATE) and measure the overall supply noise, such as with a ring oscillator. The worst-case voltage drop can be selected as an upper bound for all regions for all patterns during compaction. However, in real application, measuring the power supply noise when applying functional patterns on a tester may become expensive and impractical. Alternatively, we can indirectly specify a noise constraint upon the maximum noise-induced delay increase on all targeted paths of a vector. This approach is favored since it directly targets the cause of supply noise overkill – slow paths.

The comprehensive compaction procedure is illustrated in Figure 16. Uncompacted test patterns are loaded one by one in order and a quick pre-check is performed. This pre-check step will be discussed below. If the un-compacted test pattern exceeds the

power noise constraint, it is saved in a separate pattern list. The high power noise level of test patterns in this list is due to necessary assignments from ATPG instead of compaction. Such vectors should be very rare given the low care bit density in path delay test vectors [94]. If the power noise level for that pattern is within limits, compaction is performed. Whenever a compatible pattern in the compacted pattern list is found, a pre-check is performed to see if we can skip power supply noise estimation for this potential compacted pattern. Power noise estimation is then performed if the pre-check fails. If the supply noise level is within limits, this compaction is performed. Otherwise, the compaction is invalid and the next compatible pattern in the compacted pattern list is considered.

The pre-check step is a rough prediction of whether the test pattern has a chance to exceed the power noise limit, using the transition count in the test pattern as a noise estimator [64]. For most circuits, fewer input transitions usually imply less switching activity on chip and less power supply noise. Therefore, a transition count threshold is set by experience, so that any patterns with fewer input transitions can be assumed "safe" from exceeding the supply noise constraint. This pre-check step is extremely fast as it only scans the input test patterns without circuit simulation. In our work, the threshold is set based on our prior compaction experience. The pre-check step should not be performed if the power noise level must be guaranteed considering those rare cases where a few transitions on circuit inputs generate a large amount of switching activity or switching activity highly concentrated in a small area.

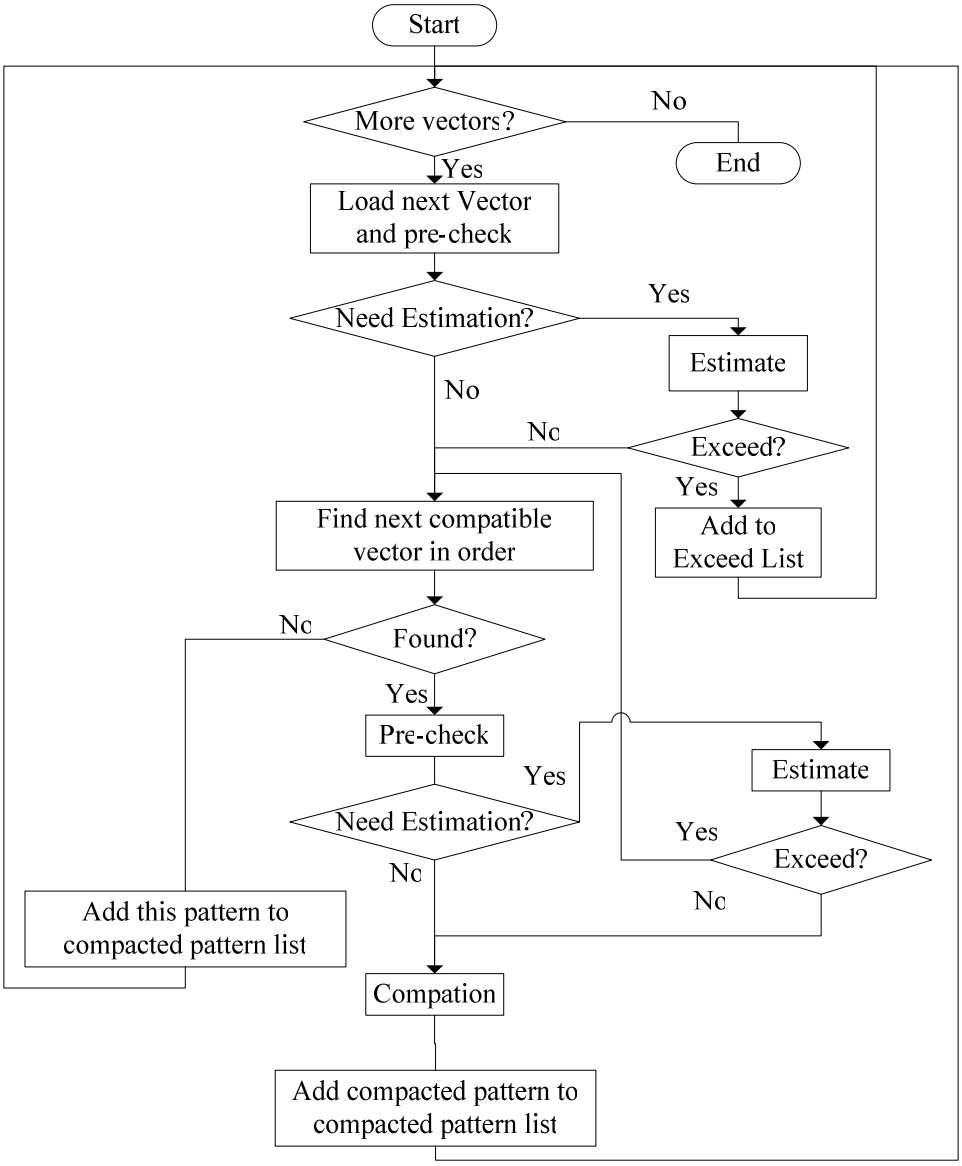

Figure 16. Flow chart of compaction with noise constraints.

4.3 Noise Consideration in Dynamic Compaction

In contrast to static compaction, dynamic compaction techniques seek to reduce test set size while tests are being generated. Dynamic compaction usually achieves a higher compaction rate than static compaction since it tends to generate test patterns with fewer

compaction conflicts with existing patterns. Various dynamic compaction techniques have been proposed [79][84][86][87][96].

A unique phenomenon in dynamic compaction is that test patterns are incomplete during compaction. This is because dynamic compaction is performed concurrently with test generation. Some of the input bits will be assigned to 0 or 1 to justify signals on gate side inputs in order to propagate the target paths, but this assignment is not unique in most cases. Therefore, as long as test generation continues, any input bits in a test pattern are still subject to change for compaction purpose, except those necessary input assignments generated by ATPG tools. As a consequence, our logic-simulation-based power supply noise analysis approach becomes inadequate to constrain noise during dynamic compaction.

We propose a different noise-constrained approach for dynamic compaction. Instead of directly analyzing power supply noise and calculating noise-induced delay, we simply estimate the average amount of switching activity in the circuit based on those necessary assignments in a test pattern and a list of signals whose values are already known (such as side inputs along the target paths). This is because the information is too limited for any more accurate circuit analysis, such that differences in switching devices (such as gate type, peak charging current and output load capacitance) can be neglected. The transition count of a circuit can roughly tell us how noisy the circuit will become. The correlation of transitions with noise-induced delay will be shown in section 5. Since logic simulation is not applicable during dynamic compaction, the expectation of transition count for the circuit is then targeted as the goal of the algorithm.

Several notations and assumptions need to be introduced before details:

1. *P{ P1 }* is the transition probability of signal line *P1*. Here the name of signal line *P1* also stands for the event that *P1* has a transition on it.
2. *P{ P0 | P1 }* is the conditional transition probability of signal line *P0*, given a transition on signal line *P1*.
3. *E{ P1 }* is the expectation of transition count on signal line *P1*.
4. *E{ P0 | P1 }* is the conditional expectation of transition count on signal line *P0* given a transition on signal line *P1*.
5. *E{ FO(P0) | P1 }* is the conditional expectation of transition count on signal *P0* and all signals in *P0*'s fanout zone, given a transition on signal line *P1*.
6. *P{ g_i }* is the conditional transition probability on gate g_i's output given a transition on one of its inputs.
7. *E{ g_i }* is the conditional expectation of transition count on gate g_i's output.
8. *E{ FO(g_i) }* is the conditional expectation of transition count on gate g_i's output and all gates in g_i's fanout zone, given a transition in one of g_i's input.
9. Circuit_Transition_Count is the expectation of transition count for the whole circuit.
10. The circuit we are dealing with is either combinational, or the combinational part of a full-scan design. In_i stands for one circuit primary input, or a scan cell output fed into the combinational part in a full-scan design.
11. Glitches are neglected.

4.3.1 Transition Probability and Expectation for Logic Gates

We first start with one logic gate to discuss its transition probability on the gate output. A 2-input AND gate is taken as an example, as shown in Figure 17.

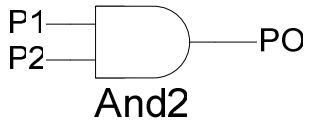

Figure 17. A 2-input AND gate.

Assume we know a transition on gate input *P1*, and that this transition has a 50% chance to be a rising transition, and 50% to be a falling one. If *P1* has a rising signal "01", *PO* is switching if and only if the *P2* signal line has a "X1" value. "X" stands for don't care. This means *P2* can be either "01" or "00" in order to get a transition on *PO*. If *P2* is "01", the same as *P1*, either of the two transitions will be propagated to *PO*, otherwise, *P2* is "11", which is a stable non-controlling value that guarantees transition propagation from *P1* to *PO*. In the same way, if *P1* has a falling transition, *P2* must have a "1X" value to make sure that *PO* will have a transition on it.

Therefore, the conditional transition probability of *PO* given a transition on *P1* is:

$P\{ PO \mid P1 \}$

$= P\{ P1 \text{ is rising} \} \cdot P\{ P2 \text{ is "X1"} \} + P\{ P1 \text{ is falling} \} \cdot P\{ P2 \text{ is "1X"} \}$

$= 50\% \cdot 50\% + 50\% \cdot 50\% = 50\%$

Since *PO* is the output of this 2-input AND gate while *P1* is the input, we also have:

$P\{ AND2 \} = P\{ PO \mid P1 \} = 50\%$

An additional assumption here is that side input signal *P2* is independent of *P1* or any other signals in the circuit. In real circuits, this independence does not exist. For example, *P2* and *P1* can be connected to each other and turn this AND gate into a buffer, such that any transition on *P1* will result in a transition on *PO* with 100% chance. However, most dependencies are much more complicated and less relevant than this example. Therefore, the dependency of side inputs on other signals is neglected in our approach to save computational cost.

We extend this conditional transition probability calculation to a 3-input AND gate, as shown in Figure 18.

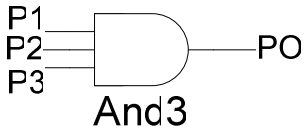

Figure 18. A 3-input AND gate.

Again, assume we know a transition on signal *P1*, and that it has 50% chance to be a rising transition and 50% to be a falling one. To guarantee a transition on *PO*, both *P2* and *P3* should have a "X1"value for a rising transition on *P1*, or a "1X" value if *P1* is falling. Therefore, the conditional transition probability of *PO*, given a transition on *P1*, becomes 25% as calculated below:

$$P\{ PO \mid P1 \}$$

$$= P\{ P1 \text{ is rising} \} \cdot P\{ P2 \text{ is "X1"} \} \cdot P\{ P3 \text{ is "X1"} \}$$

$$+ P\{ P1 \text{ is falling} \} \cdot P\{ P2 \text{ is "1X"} \} \cdot P\{ P3 \text{ is "1X"} \}$$

$= 50\% \cdot 50\% \cdot 50\% + 50\% \cdot 50\% \cdot 50\% = 25\%$

Which means:

$P\{ AND3 \} = 25\%$

Similar to the 2-input AND gate, the two side input signals *P2* and *P3* are both assumed independent of P1 or any other signals in the circuit.

Once we get the conditional transition probability of a gate, we are able to determine the conditional expectation of transition count on the gate output. For example, on the 2-input AND gate, the conditional expectation of transition count on the gate output is:

$E\{ AND2 \} = E\{ PO \mid P1 \} = P\{ PO \mid P1 \} = 0.5$

A similar calculation is performed on all gate types to find $P\{ g_i \}$ and $E\{ g_i \}$. The results are shown in Table 3.

Table 3. Transition probability and expectation for logic gates.

Gate Type	Inputs	$P\{g_i\}$	$E\{g_i\}$
(N)AND	2	50%	0.5
	3	25%	0.25
	4	12.5%	0.125
(N)OR	2	50%	0.5
	3	25%	0.25
	4	12.5%	0.125
INVERTER	1	100%	1
BUFFER	1	100%	1
X(N)OR	2	50%	0.5
	3	25%	0.25
	4	12.5%	0.125

4.3.2 Pre-compaction Analysis

Based on the $P\{ g_i \}$ and $E\{ g_i \}$ calculation introduced in the previous section, we are able to compute $E\{ FO(In_i) \mid In_i \}$ for each In_i, which is the conditional expectation of transition count for a combinational circuit given a transition on one input. An example is shown in Figure 19, which consists of four 2-input AND gates.

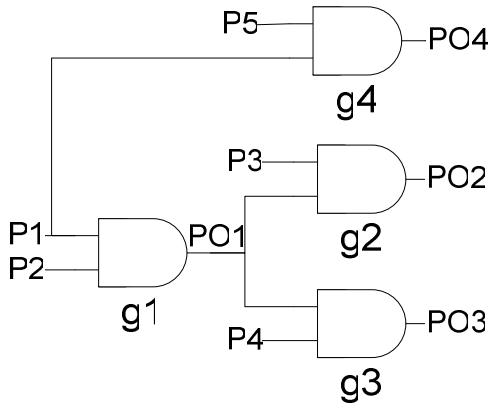

Figure 19. A circuit for analysis.

We have:

$E\{ FO(g2) \} = E\{ g2 \} = P\{ g2 \} = 0.5$

$E\{ FO(g3) \} = E\{ g3 \} = P\{ g3 \} = 0.5$

Then we look at the fanout zone of gate *g1*. Given a transition on *P1*, the conditional expectation of transition count in *g1*'s fanout zone is:

$E\{ FO(g1) \}$

$= E\{ PO1 \mid P1 \} + E\{ PO2 \mid P1 \} + E\{ PO3 \mid P1 \}$

$= P\{PO1 \mid P1 \} + P\{ PO2 \mid PO1 \} * P\{PO1 \mid P1 \} + P\{ PO3 \mid PO1 \} * P\{PO1 \mid P1 \}$

$= P\{ g1 \} * (1 + E\{ FO(g2) \} + E\{ FO(g3) \})$

This equation shows that $E\{ FO(g_i) \}$ for a gate can be calculated using $P\{ g_i \}$ for this gate and the value of $E\{ FO(g_k) \}$ for any direct fanout gate g_k.

We then calculate $E\{ FO(In_i) \mid In_i \}$ in the same way. Once we get $E\{ FO(g_i) \}$ for any direct fanout gate g_i for input In_i, say *P1* for the circuit in Figure 19, we have:

$E\{ FO(P1) \mid P1 \} = E\{ FO(g4) \} + E\{ FO(g1) \}$

This is slightly different from the previous calculation. However, we can view each input as a pseudo-buffer gate which has 100% conditional transition probability, and that the transition on this pseudo-buffer gate does not need to be counted. Thus, $E\{ FO(In_i) \mid In_i \}$ for each In_i is the sum of $E\{ FO(g_i) \}$ for all direct fanout gates of In_i.

A recursive algorithm was developed based on this calculation to compute $E\{ FO(In_i) \mid In_i \}$ for all In_i. The goal is to compute the average noise level in the circuit given a transition on one circuit input, while test pattern information is not available and logic simulation is not applicable.

The algorithm is shown in Figure 20. Starting from every circuit input, the recursive algorithm is applied to compute the conditional expectation of transition count in the fanout zone for each gate in the circuit, and eventually for each inputs. Each gate in the circuit will be visited once in this algorithm to calculate $P\{ g_i \}$ and $E\{ FO(g_i) \}$.

This pre-compaction analysis works well for fan-in-free circuits. For circuit with fan-in, $E\{ FO(g_i) \}$ can be counted more than once. Take Figure 21 as an example, $E\{ FO(g3) \}$ is included both in both $E\{ FO(g2) \}$ and $E\{ FO(g4) \}$, hence it is counted twice in $E\{ FO(g1) \}$. This is reasonable since $E\{ FO(g3) \}$ can come from a transition on either *g2* or *g4*. However, the transition count due to transitions on both gate inputs

are counted twice, which leads to an over-estimation of $E\{ FO(g1) \}$. The impact of this over-estimation factor on the algorithm will be evaluated in the experiments.

```
PRE_COMPACTION_CIRCUIT_ANALYSIS(circuit)
{
    For each circuit input In_i
    {
        For each direct fanout gate g_i of input In_i
            STATIC_FANOUT_ANALYSIS(g_i)
        E{ FO(In_i) | In_i } = Σ E{ FO(g_i) }
    }
}


STATIC_FANOUT_ANALYSIS (g: the gate under analysis)
{
    Look up P{ g } by gate type
    For each direct fanout gate g_i of gate g
    {
        if g_i has not been visited before
            STATIC_FANOUT_ANALYSIS (g_i)
    }
    E{ FO(g) } = P(g) · (1 + Σ E{ FO(g_i) })
}
```

Figure 20. Pre-compaction analysis of the circuit to compute the average transition count given a transition on each circuit input.

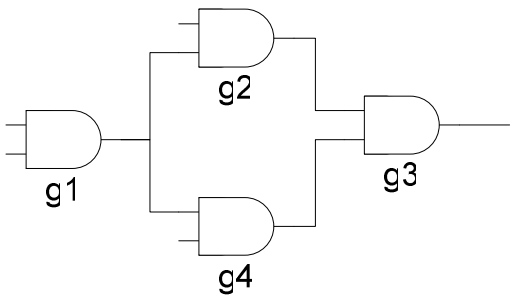

Figure 21. A circuit with fan-in.

4.3.3 Transition Count Prediction during Compaction

During dynamic compaction of path delay tests, we have values on a list of external and internal signals, such as circuit input and internal gate output, based on path justification [94]. Compared to pre-compaction analysis, this additional information can help us to get a more accurate estimation for average transition count in the circuit. We propose a second algorithm to compute the expectation of transition count in the whole circuit, denoted as *Circuit_Transition_Count*, given values of a number of internal and external signals.

The pre-compaction circuit analysis, as introduced in the last section, must be applied before this algorithm. Hence, every gate in the circuit will have an initial $P\{ g_i \}$ and $E\{ FO(g_i) \}$ and every circuit input In_i will have an initial value of $E\{ FO(In_i) \mid In_i \}$ before this second algorithm is applied. Then each signal whose value is known will be reviewed to update $E\{ FO(g_i) \}$ and $E\{ FO(In_i) \mid In_i \}$ for relevant gates and circuit inputs to get a more accurate *Circuit_Transition_Count*.

Assume we know a gate whose output signal has a transition on it. The value of $P\{ g_i \}$ for this gate, whatever the gate type, should be updated to 1. Consequently, the value of $E\{ FO(g_i) \}$, should also be corrected by dividing the old value of $P\{ g_i \}$ into it.

The updated $E\{ FO(g_i) \}$ of this gate should be added to *Circuit_Transition_Count*. In the meantime, the old value of $E\{ FO(g_i) \}$ should be properly removed from $E\{ FO(g_i) \}$ of its predecessor gates (gates whose fanout zone covers this gate) and $E\{ FO(In_i) \mid In_i \}$ of predecessor circuit inputs. A recursive function is needed for this purpose. For instance, in Figure 19, if *PO2* is known to have a transition on it, $E\{ FO(g2) \} \cdot P\{ g1 \}$

should be subtracted from $E\{ FO(g1) \}$. And $E\{ FO(P1) \mid P1 \}$ should also be updated accordingly.

In the same way, if a signal is known to have a stable value, the value of $P\{ g_i \}$ for this gate should be updated to 0, and the conditional expectation will also become 0. Similarly, the old value of $E\{ FO(g_i) \}$ should be properly removed from $E\{ FO(g_i) \}$ of its predecessor gates and $E\{ FO(In_i) \mid In_i \}$ of relevant circuit inputs.

Sometimes the output values of two gates are known, while one gate is in the fanout zone of the other. Possibly the gates in their common fan-in zone may get updated twice. To improve algorithm efficiency in updating relevant $E\{ FO(g_i) \}$ and $E\{ FO(In_i) \mid In_i \}$, once we find a gate whose value is known, we will update its value and stop searching its fan-in zone.

Once all signals with known values have been processed, the $E\{ FO(In_i) \mid In_i \}$ associated with each circuit input has been updated to a more accurate value. However, to get *Circuit_Transition_Count*, we still need to know which inputs are switching. Since the input pattern we have is incomplete and undetermined, we need to design a method to determine which input signals are switching and add their $E\{ FO(In_i) \mid In_i \}$ to *Circuit_Transition_Count*.

To solve this problem, we first list all gates that are known to have a transition on their outputs. The fan-in zone for each of these gates is analyzed to find a list of circuit inputs that may produce the transition on this gate. These circuit inputs are called fan-in inputs for this gate. Obviously, at least one of these fan-in inputs should be switching to validate the transition on this gate output. Different strategies can be adopted in selection

of the switching fan-in inputs.

A conservative method is to select the fan-in input with minimum $E\{ FO(In_i) \mid In_i \}$ for each gate with a transition on its output, in case none of the fan-in inputs have been selected before or are known to be switching. This method should underestimate *Circuit_Transition_Count* in most cases. The aggressive method, in contrast to the conservative one, is to select the fan-in input with maximum $E\{ FO(In_i) \mid In_i \}$ for each gate with a transition on its output, or even select all fan-in inputs that may produce this transition. This method is likely to overestimate *Circuit_Transition_Count*. These approaches will be compared in experiments.

The algorithm is shown in Figure 22. For simplicity, we make a pseudo-buffer gate for each circuit input In_i. This gate is named as gIn_i. $P\{ gIn_i \}$ is 100%, and $E\{ FO(gIn_i) \}$ equals $E\{ FO(In_i) \mid In_i \}$ of its corresponding In_i.

In this algorithm, one major source of error lies in the process of determining switching fan-in inputs, since the variation of $E\{ FO(In_i) \mid In_i \}$ from input to input may be large. Therefore, different selection strategies will be compared in the experiments to evaluate their accuracy and efficiency.

```
TRANSITION_COUNT_ESTIMATION (circuit, a list of gates of known value g1-m)
{
  For each gate of known value gi (excluding pseudo-gates of primary inputs)
  {
    UPDATE_FANIN (gi, 0)
    Add E{ FO(gi) } to Circuit_Transition_Count
  }
  For each switching gate gi
  {
    Select fan-in inputs
    Add their transition_count to Circuit_Transition_Count
  }
}

RECURSIVE_UPDATE_FANIN (gate g, change in g's fanout transition count δ )
{
  If g is a circuit input gIni
    Update E{ FO(gIni) } with δ
  If g's value is known and g is not the gate that starts the recursive process
    Update E{ FO(g) } with δ
  If g is the gate that starts the recursive process
  {
    Keep the old value of E{ FO(g) } as δE
    Update P{g} and E{ FO(g) }
    For each of g's fanin gates gi
      RECURSIVE_UPDATE_FANIN(gi, - δE)
  }
  Else
  {
    Update E{ FO(g) } with δ
    For each of g's fanin gates gi
      RECURSIVE_UPDATE_FANIN(gi, P{g} · δ)
  }
}
```

Figure 22. Circuit transition count prediction algorithm

Another source of error is duplicate count for multiple transitions on gate inputs. This is similar to the problem for fan-in circuits that we discussed in the previous section. However, the problem may occur for a fan-in-free circuit as well. Take the 2-input AND gate in Figure 17 for example. Assume this is the only gate in the circuit, so *P1* and *P2* are both circuit inputs while *PO* is the output. *E{ FO(AND2) }* is included in both *E{ FO(P1) | P1 }* and *E{ FO(P2) | P2 }*. Assume both *P1* and *P2* are known or selected to be switching signals, *Circuit_Transition_Count* would be the sum of *E{ FO(P1) | P1 }* and *E{ FO(P2) | P2 }*. Thus, *E{ FO(AND2) }* is counted twice in *Circuit_Transition_Count*. This will likely lead to over-estimation of transition count for the entire circuit.

4.4 Test Fill

As we discussed in the compaction sections, the ATPG tool only needs to specify values on a subset of all primary inputs (and all scan flip-flop outputs for scan designs), and leave the rest as don't care. Compaction will set some of the don't care bits to either "0" or "1". Algorithms can be applied to assign specific logic values to don't care bits. This procedure is called test fill. Test data compression techniques using simple on-chip decoding hardware and by compressing the don't care bits are widely used to reduce test data volume and test time. The on-chip decoding hardware will also fill the don't care bits after decompression is performed.

A widely used test fill technique is random fill. In industry, random fill of don't care bits is usually applied to delay test patterns to increase fortuitous detection of non-target

defects. Unfortunately, random fill can produce excessive supply noise and result in overkill [23]. A random fill technique will randomly set each don't care bit to either "0" or "1". In delay testing, random fill usually leads to a lot of switching activity on the circuit.

Another popular test fill technique for delay test patterns is minimum-transition fill. This technique targets minimizing transitions on input test patterns. For instance, a bit with a "0X" value will be assigned to "00", and a "X1" bit with be set to "11". It is expected that delay test patterns with minimum-transition fill tend to create a low-noise environment for the propagation paths.

In addition, we can also apply 0-fill to test patterns, which sets every don't care bit to 0; and 1-fill that sets every don't care bit to 1. Unlike the previous two techniques, it is hard to tell whether these techniques will make a circuit more noisy or not. It largely depends on specific test patterns and circuit design.

The test fill technique can also be weighted with a specific value. For instance, in random fill, a don't care bit can be assigned to 0 with some probability other than 50%. If the probability is 90%, the filled test patterns will be quite close to the ones with 0-fill.

Test fill has a significant impact on circuit noise. Comparison of test fill techniques will be included in section 5.

5. EXPERIMENTS

Experiments have been conducted to validate the two noise models proposed in section 3, and to show the noise impact on compaction and test fill. Two sets of experiments have been performed. The first set of experiments is based on Noise Model I, and the measurements are taken on ISCAS89 benchmarks. In additional to model validation, we have also performed experiments on static compaction and test fill using Noise Model I. The experiments on transition count prediction, which was introduced in section 4, have also been performed under the same framework. All the results are included in section 5.1.

The second set of experiments, based on Noise Model II, was conducted on a wire bond industry design during an internship in Philips Research Lab, Eindhoven, the Netherlands, in summer 2005. This Research Lab now belongs to NXP Semiconductors. The experiments mainly focused on validation of Noise Model II. Compaction experiments have not been performed due to limited resources. Some supplementary silicon data was also collected on this design. All these results are included in section 5.2.

The comparison of the two models will be discussed in the last section.

5.1 Experiments on ISCAS Benchmark Circuits

The first set of experiments was performed on three ISCAS89 benchmark circuits, s1488, s38417 and s35932. The layout of these benchmark circuits was generated with Cadence Silicon Ensemble using TSMC 180 nm, 1.8 V technology.

We assume area-array bonding for the circuit package. For all three circuits, the power grid design has only one pad, which is located in the center of the circuits. This is because these circuits are relatively small. In industrial designs, the number of logic gates covered by each power pad is usually larger than the total number of gates for each of these three benchmark circuits.

Delay test pattern sets were generated using the CodGen path delay test generator [94]. Robust launch-on-capture path delay tests targeting the longest rising and falling transition path through every line in the circuit (termed KLPG-1) were generated. One path is targeted per pattern. For noise model validation purposes, only selected test patterns from the original pattern sets were used in the experiments to reduce irrelevant error. For static compaction and test fill purposes, however, the entire test sets generated by CodGen were used. The transition count prediction algorithm uses both original test sets and compacted test sets for experiments, the latter coming from static compaction experiments.

5.1.1 Validation Results for Noise Model I

Two benchmark circuits, s1488 and s38417, were used in our experiments to validate Noise Model I. The circuit s38417 has over twenty thousand logic gates, while the circuit s1488 has less than a thousand.

For noise model validation purposes, a number of test patterns are selected from the complete test set generated by CodGen. The target paths of those selected test patterns must be strictly robust paths with side inputs fixed at non-controlling values. This requirement guarantees that the signals are propagated on the exact paths. In addition to

this, these static sensitized paths are also free from glitches, as observed during the experiments. This is because glitches may cause irrelevant delay error during analysis. The “don’t care” bits of these selected patterns are filled with minimum transition for s1488, as introduced in section 4, and random fill for s38417, to generate a certain amount of switching activity in the circuit.

Noise Model I, which has been implemented in our analysis tool for these experiments, models off-chip current as a constant current source that averages the previous K clock cycles of current consumption. Details of off-chip current modeling were introduced in section 3. Dynamic charging/discharging current is calculated using simulation-based table method. Short circuit current is calculated using table method as well. Input slope was computed by Static Timing analysis for switching current calculation that mentioned above.

The delay modeling approach we adopt in these experiments models both delay and transition time as a function of input slope, output capacitive load and device voltage level. Again, simulation-based table method is used. In addition to this, temporal voltage variation is also considered in delay calculation. The supply voltage drop for each gate, when its transition occur, is approximated as a fraction of worst-case voltage drop based on nominal propagation time. Spatial voltage variation is neglected in delay calculation as mentioned in section 3.

5.1.1.1 Circuit s1488

Forty test patterns that target glitch-free static sensitized paths have been selected for circuit s1488. We first evaluate the error of predicted voltage drop compared with circuit

simulation results from Cadence Spectre, then present the comparison of predicted and simulated nominal delay to evaluate the error of the nominal delay model, and eventually compare noise-induced delay between our analysis and simulation.

In Figure 23, we plot the correlation between simulation and our voltage drop analysis for these 40 patterns on circuit s1488. The correlation $R^2 = 0.9319$ with non-zero intercepts. Our analysis tool tends to slightly overestimate voltage drop when the actual voltage drop is approximately less than about 9%, and slightly underestimate when the actual voltage drop is higher. The main reason is that the short circuit current is quite sensitive to the input slope. We generated input slopes with an STA tool that searches for worst-case delay. This results in larger input slopes and more short circuit charge consumption in our analysis than in circuit simulation. However, when the voltage drop becomes larger in simulation, more unexpected noise, such as more glitches due to slower speed, may appear on chip and result in more current drawn from the power supply network.

Then we look at the voltage error for each vector. The results are shown in Figure 24. The voltage error for a logic gate is defined as the difference between the worst-case voltage level from our analysis tool and from Spectre simulation divided by the nominal voltage (1.8V in our experiments). Hence, the voltage error for each test is the maximum error among all the gates on its targeted path. Our experimental results show that the voltage error is within -1.5% to 1.7%, with an average of 1%, while the actual voltage drop varies from 3% to 11%. In addition, the voltage error shows that our analysis tool tends to slightly overestimate voltage drop when the actual voltage drop is less than

about 9%, and slightly underestimate when the actual voltage drop is higher than 9%.

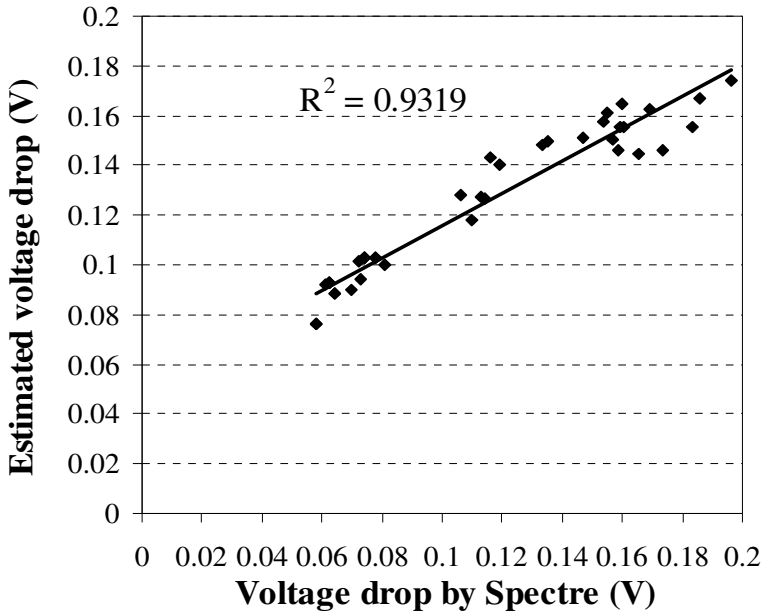

Figure 23. Correlation of voltage drop on s1488.

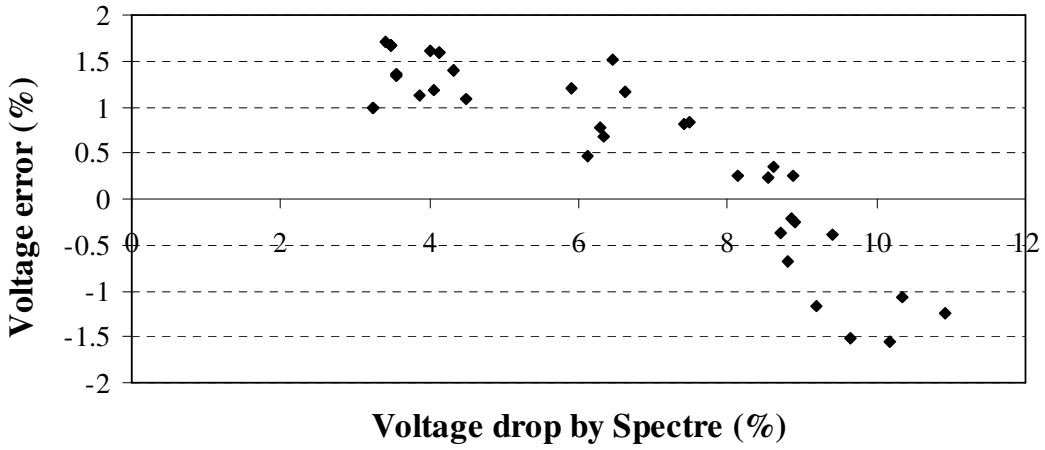

Figure 24. Voltage error of s1488.

Our second step is to evaluate the accuracy of the delay model without supply noise.

We calculate nominal path delay using our analysis tool by assuming nominal voltage level for all logic gates. During Spectre simulation, the supply voltage pins of logic gates are all connected directly to an ideal voltage source to eliminate noise and simulate nominal path delay. The correlation of nominal path delay data between simulation and our analysis tool is presented in Figure 25. The correlation is $R^2 = 0.9989$ with zero y-intercept. This means the nominal delay model we used in the experiments is quite accurate compared with simulation data.

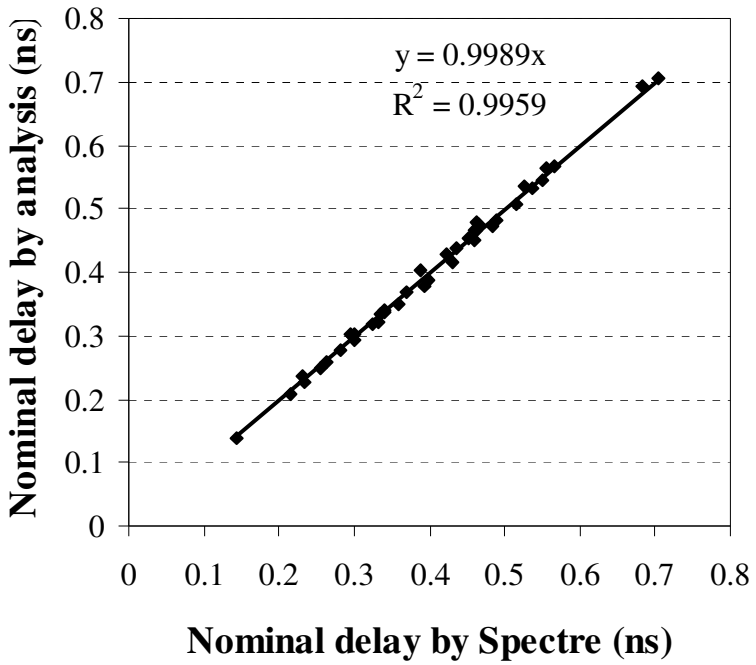

Figure 25. Correlation of nominal path delay for circuit s1488.

Similar to voltage error, delay error is defined as the error of analysis relative to the simulated nominal path delay. The nominal delay error over the 40 patterns is approximately -4% to 4% with an average of 1.8%, as shown in Figure 26. The error

distribution is independent of path delay. However, the error when the nominal path delay is small is usually quite large. This is because when nominal path delay is small, delay error is more sensitive to the absolute difference between analysis and simulation.

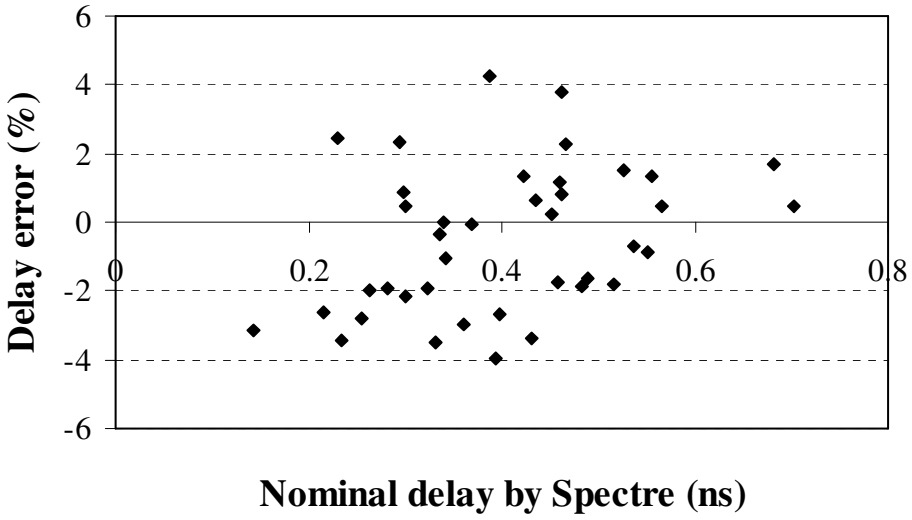

Figure 26. Nominal delay error of s1488.

The noise-induced delay is then calculated using our power supply noise analysis tool, and is also simulated by Cadence Spectre with complete specification of layout and power supply network parasitics. We show the correlation between analysis and simulation of s1488 with $R^2 = 0.9951$ and intercept = 0, in Figure 27. The path delay error is shown in Figure 28. With power supply noise, path delay error is -3% to 6%, with an average of 1.9%. Again, we find the error is larger when the nominal path delay is smaller, due to higher sensitivity of delay error to the absolute difference when path delay is smaller.

This error of noise-induced delay is only slightly larger than nominal delay error, so

most of the error is due to the delay model. On the other hand, the noise-induced delay error is much smaller than the voltage error shown previously. One reason is that the noise-induced delay calculation takes temporal voltage variation into consideration. For logic gates whose transition time is early, the effective voltage drop is smaller than worst-case voltage drop, which reduced the voltage drop impact on delay, and consequently reduced delay error due to voltage error. Another reason is that the gate delay sensitivity to supply voltage variation in the generated gate library and physical design for s1488 is low. As observed in the experiments, a 1% change in gate supply voltage level often causes less than a 1% change in gate delay variation. The designs that are more sensitive to voltage variations should be used in future work.

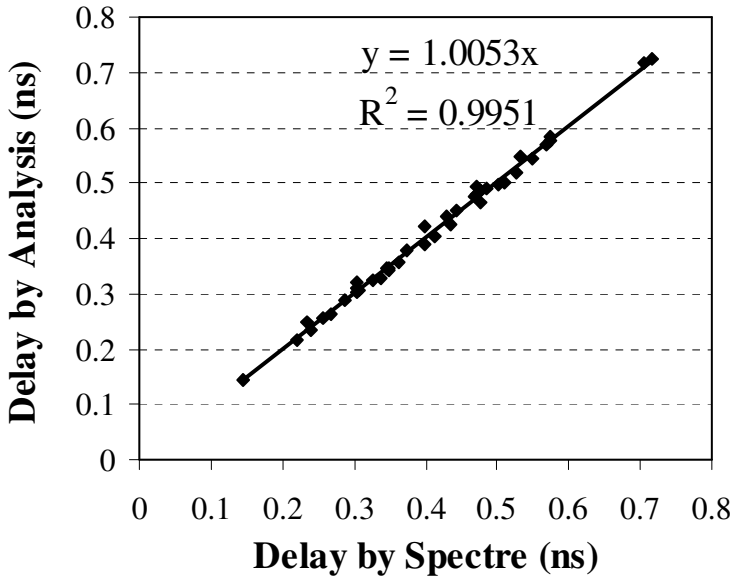

Figure 27. Correlation of path delay with supply noise on s1488.

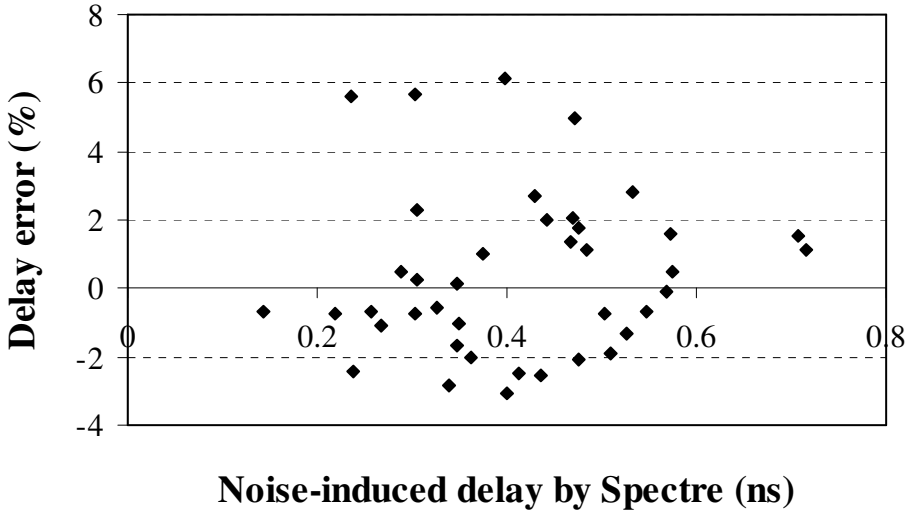

Figure 28. Delay error of noise-induced delay of s1488.

5.1.1.2 Circuit s38417

Similar experiments have been performed on circuit s38417 to evaluate Noise Model I. Circuit s38417 contains 22K gates and is much larger than s1488. Data for only 16 test patterns has been generated due to the high cost of Spectre simulation. These 16 test patterns target glitch-free static sensitized paths. As with s1488 experiments, we first evaluate the error of voltage drop compared with Spectre simulation, and then compare noise-induced delay between our analysis and simulation. The error of nominal delay models was not analyzed, since it was done for circuit s1488.

The correlation of voltage drop for circuit s38417 is shown in Figure 29. R^2 is 0.9723 with zero y-intercept. This correlation is much better than s1488, which has $R^2 = 0.9319$ with non-zero y-intercept, as shown in Figure 23. One explanation is the averaging effects of larger circuits. We also found that the voltage drop level for these 16 patterns

on s38417, measured by Spectre simulation, varies from 7% to 17% of nominal voltage level, with an average of 12%. This is higher than the voltage variation region for circuit s1488, which was 3% to 11% of nominal voltage with an average of 6%. Therefore, another explanation is that this voltage drop analysis tends to be more accurate when the circuit is not so quiet.

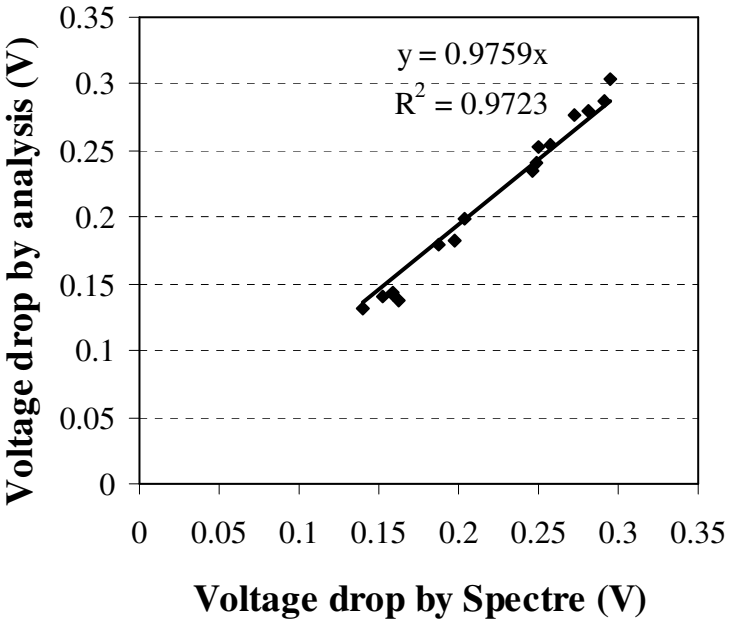

Figure 29. Correlation of voltage drop on s38417.

The voltage error for circuit s38417 is shown in Figure 30. For s38417, experimental results show that the voltage error is -1.4% to 0.6%, with an average of -0.4%, while the actual voltage drop varies from 7% to 17%. This is better than the voltage error for s1488.

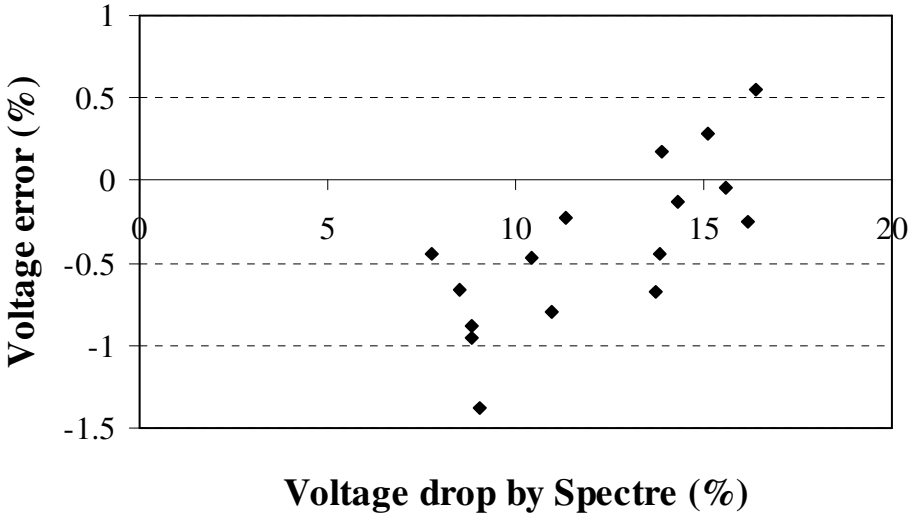

Figure 30. Voltage error for circuit s38417.

An interesting observation for the voltage error distribution in Figure 30 is that our analysis tool tends to overestimate power supply noise when actual voltage drop becomes large (larger than 15% in this case). We compare it with the voltage error of s1488 shown in Figure 24, which has less noise and voltage drop. For s1488, our analysis tool tends to overestimate when simulated voltage drop is less than 9%, and underestimate when simulated voltage drop becomes larger than 9%. If we put these data together, we can see that the analysis tool tends to underestimate when applied to circuits with medium noise level, and overestimate if circuits are too quiet or too noisy.

Nominal delay error for circuit s38417 is neglected in the analysis as explained in the beginning of the section.

The noise-induced delay was then calculated using our power supply noise analysis tool, and compared with simulated results from Spectre simulation. We have $R^2 = 0.952$

with zero y-intercept, in Figure 31. This is not as good as the correlation for s1488. The path delay error is also larger than s1488. However, a larger test pattern sample size may provide a more convincing conclusion.

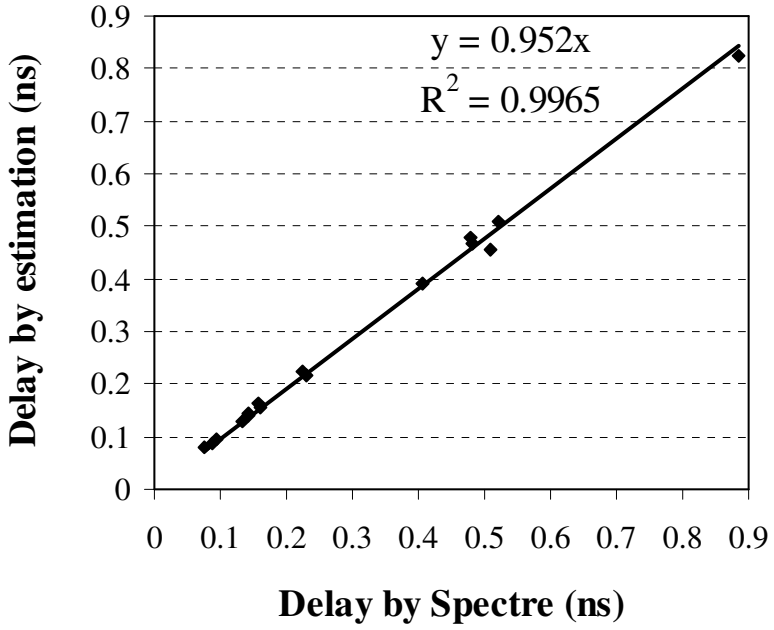

Figure 31. Correlation of path delay with supply noise on s38417.

5.1.2 Static Compaction Results

A static test pattern compaction tool was developed to carry out experiments of noise constrained static compaction. The power supply noise analysis tool is based on Noise Model I, as validated in the previous section, and was integrated into the compaction tool. The algorithm of noise constrained static compaction was introduced in section 4. The tool was written in C++ and runs on a 2.3 GHz Pentium 4 system.

The experiments were performed on ISCAS89 benchmark circuit s38417. Path delay

test sets used for compaction were generated by CodGen [94]. They are robust tests using launch-on-capture targeting the longest rising and falling transition path through every line in the circuit (termed KLPG-1). All don't care bits of the patterns are reserved for compaction. Once we need to evaluate power supply noise and noise-induced delay for a test pattern with don't care bits, these don't care bits are filled with minimum transition so as not to introduce extra noise.

As discussed in section 3, the on-chip decoupling capacitance will affect voltage drop. Ratio γ is defined as the on-chip power grid capacitance divided by the total signal net capacitance of the circuit. In our experiments, we set γ to 3.8 so as to keep voltage drop typical.

The initial voltage of each cycle is set to the nominal voltage level, which is 1.8 V in our experiments, assuming the supply voltage returns approximately to the nominal value at the start of the cycle. Since off-chip current in Noise Model I is the average current consumed in the previous K cycles, we arbitrarily set it to zero, simulating the typical *Ldi/dt* problem of scan delay test, in which circuit switching dies down during the long scan-to-launch delay.

If we perform a static forward-order compaction without noise analysis, the resulting test set (denoted as s) can serve as an approximate lower bound for any compaction method that considers power supply noise. The un-compacted test set (denoted as ucs) for this benchmark contains 13,941 vectors (ucs = 13,941) and has a fill-rate of 2.5%. For this test set, s is 940 with a fill-rate of 25%, and the static forward-order compaction without noise for s38417 takes 95 seconds. The pre-check procedure has been

implemented here by setting the transition count threshold to 0.1%, based on experience. For s38417, a transition count less than 0.1% means that there is only one transition on all input pins and scan chains. Experiments show that none of those patterns exceed even the tightest voltage drop constraint we have ever applied.

As mention in section 3, two kinds of constraints on power supply noise have been implemented. One is worst-case voltage drop in the circuit, while the other is maximum percentage increase of path delay caused by power supply noise. Since delay is the major concern of the path delay test, it is the eventual estimate of the power supply noise effect on delay testing.

Table 4 shows how the compaction results vary with worst-case voltage drop constraint. The first column in Table 4 shows which type of constraint is applied. Here we use worst-case voltage drop in the circuit as constraint. Column 2 is the percentage of constraint applied in the experiments. Column 3 is the number of test patterns that exceeds the noise constraint prior to compaction even filled with minimum transition, and column 4 is the size of the compacted test set excluding any original failed test patterns (patterns whose noise level is too high prior to compaction, as listed in column 3). Column 5 lists α, which is the percentage increase in compacted test set size due to the noise constraint. Column 6 is the number of times that the noise analysis procedure is skipped through the pre-check step, and column 7 is the total number of calls to the power noise analysis procedure during compaction. Column 8 lists the failure ratio β, the fraction of the times that a potential vector compaction exceeds the noise constraint. Column 9 is the fill-rate of test patterns after compaction.

Table 4. Compaction results for s38417 with worst-case voltage drop constraint (The fill-rate of the un-compacted test set is 2.5%, with ucs = 13941 and s = 940).

Constraint Type	Constraint (%)	Initially Failed Patterns	Compacted Test Size	α (%)	Skipped Calls by Pre-check	Analysis Calls	β (%)	Compacted Fill-rate
Worst-case Voltage Drop	3	1,265	1,168	158.8	2,797	544,141	95.8	0.107
	4	610	1,020	73.4	2,798	187,620	87.5	0.153
	5	139	947	15.5	2,797	48,294	50.3	0.221
	7.5	0	940	0	2,798	24,148	0.02	0.250
	10	0	940	0	2,798	24,144	0	0.250

Table 5 shows running time of noise constrain compaction whose results are presented in Table 4. The first 4 columns contain data from Table 4. Column 1 and column 2 are exactly the same as the first two columns of Table 4, which shows the type of noise constraint and the percentage of constraint, respectively. Column 3 shows the same data as column 6 in Table 4, which is the number of times that the noise analysis procedure is skipped through pre-check step, and column 4 is the same as column 7 in Table 4, which is the total number of calls to the power noise analysis procedure during compaction. The last four columns show the run time. Column 5 is the total time spent on logic simulation, column 6 is the total time spent on noise estimation, which includes the logic simulation time. Column 7 shows the total CPU time, which consists of both noise analysis and compaction time. The last column shows the average CPU time per analysis call. There is no prior work in the literature that can be used for comparison.

Table 5. Compaction running time for s38417 with worst-case voltage drop constraint.

Cons-traint Type	Cons-traint (%)	Skipped Calls by Pre-check	Analysis Calls	Logic Simula-tion Time	Noise Analysis Time	CPU Time	Time Per Analy-sis(ms)
Wors t-case Vol-tage Drop	3	2,797	544,141	7hr 16min	12hr 34min	12hr 34min	83.1
	4	2,798	187,620	2hr 39min	4hr 37min	4hr 37min	88.6
	5	2,797	48,294	39min	1hr 8min	1hr 8min	84.5
	7.5	2,798	24,148	19min	34min	34min	84.4
	10	2,798	24,144	19min	35min	35min	89.6

Generally, the tighter voltage drop constraint results in a larger compacted test set. We also found in experiments that the total CPU time is only slightly larger than noise analysis time, since the compaction time is relatively insignificant compared with noise estimation. Thus, a tighter constraint requires more analysis calls and more CPU time.

We also found that the increase of compacted test set size is relatively small compared with the increase in estimation calls. Note that in Table 4, when a 3% worst-case voltage drop constraint is applied, β equals 95.8%, meaning only 4.2% of compaction trials are approved, while α is 158.8%, which means the compacted test set size is about 2.5 times larger than s. The main reason is that the fill-rate of un-compacted patterns is quite low (2.5% in our experiments), and each pattern may have many compatible patterns. Therefore, most test patterns will finally get compacted after a number of trials. In other words, our compaction tool tends to compact vectors in a way

that power supply noise is more evenly distributed among all compacted test patterns.

The pre-check step speeds up the whole process by reducing the total number of analysis calls. However, since the transition count threshold we set in the experiments was only 0.1%, not many noise analysis calls were skipped during pre-check. A larger transition count threshold would further speed up the whole process, but at the risk of missing patterns that exceed the noise constraint. Note that the number of skipped analysis calls due to pre-check hardly changes with the constraint, mainly because the transition count threshold set in the experiment is quite tight, and most compaction will generate at least one more transition and exceed the threshold. Therefore, most of these skipped analysis calls come from initial patterns.

We show the compaction results with maximum path delay increase constraint in Table 6, and the corresponding compaction time in Table 7. The data definition in each column of Table 6 and Table 7 are the same as Table 4 and Table 5, respectively.

Table 6. Compaction results for s38417 with max path delay increase constraint (The fill-rate of the un-compacted test set is 2.5%, with ucs = 13941 and s = 940).

Cons-traint Type	Cons-traint (%)	Initially Failed Patterns	Com-pacted Test Size	α (%)	Skipped Calls by Pre-check	Ana-lysis Calls	β (%)	Com-pacted Fill-rate
Max Delay Increase	3	916	958	99.4	2,920	276,906	91.7	0.132
	5	265	947	28.9	2,841	129,810	81.6	0.198
	7.5	86	937	8.8	2,803	49,763	51.7	0.231
	10	17	938	1.6	2,800	38,109	36.7	0.247
	15	0	940	0	2,798	24,145	0	0.250

Table 7. Compaction running time for s38417 with max path delay increase constraint.

Cons-traint Type	Cons-traint (%)	Skipped Calls by Pre-check	Analysis Calls	Logic Simula-tion Time	Noise Analysis Time	CPU Time	Time Per Analy-sis(ms)
Max Delay Increase	3	2,920	276,906	3hr 50min	6hr 33min	6hr 34min	85.1
	5	2,841	129,810	1hr 48min	3hr 6min	3hr 7min	86.1
	7.5	2,803	49,763	41min	1hr 12min	1hr 12min	87.0
	10	2,800	38,109	32min	57min	57min	88.9
	15	2,798	24,145	20min	36min	36min	89.4

As with the voltage constraint, the path delay increase constraint also results in a larger compacted test set. Thus, a tighter constraint requires more analysis calls and more CPU time. Delay constraints further increase running time due to the need to estimate the delay of every target path of the test pattern. We find that data in Table 6 and Table 7 are consistent with results shown in Table 4 and Table 5. However, when path delay increase is constrained, the running time per analysis call is slightly increased due to the path delay calculation.

Table 8 shows the compaction results when on-chip decoupling capacitance varies. Table 9 lists the running time of these experiments. The constraint applied in the experiments is 10% worst-case voltage drop. The ratio γ, which was defined earlier, is the on-chip power grid capacitance divided by the total signal net capacitance of the circuit. Larger values of γ are obtained by increasing on-chip decoupling capacitance.

The data in Table 8 again proves that decoupling capacitance, which, in our model, is the main provider of charge at the early stage of the cycle, has a dominating impact on voltage drop and path delay.

Table 8. Compaction results for s38417 when decoupling capacitance varies with worst-case voltage drop constrained at 10% (The fill-rate of the un-compacted test set is 2.5%, with ucs = 13941 and s = 940).

γ	Initially Failed Patterns	Compacted Test Size	α (%)	Skipped Calls by Precheck	Analysis Calls	β (%)	Compacted Fill-rate
1.2	677	1,024	81.0	2,797	207,651	88.74	0.147
1.5	198	956	22.8	2,798	63,593	62.37	0.209
2.3	0	940	0	2,798	24,717	23.18	0.250
3.1	0	940	0	2,798	2,4148	0.02	0.250
3.9	0	940	0	2 798	24,144	0	0.250

Table 9. Compaction running time for s38417 when decoupling capacitance varies with worst-case voltage drop constrained at 10%.

γ	Skipped Calls by Pre-check	Analysis Calls	Logic Simulation Time	Noise Analysis Time	CPU Time	Time Per Analysis (ms)
3	2,920	276,906	3hr 50min	6hr 33min	6hr 34min	85.1
5	2,841	129,810	1hr 48min	3hr 6min	3hr 7min	86.1
7.5	2,803	49,763	41min	1hr 12min	1hr 12min	87.0
10	2,800	38,109	32min	57min	57min	88.9
15	2,798	24,145	20min	36min	36min	89.4

5.1.3 Test Fill Results

Three ISCAS89 benchmarks, s1488, s38417 and s35932, were used for experiments on noise impact in test fill. The pattern sets used in test fill experiments are un-compacted test sets generated by CodGen. These patterns can be either randomly filled or minimum transition filled to generate high or low noise levels. The care bit density of each un-compacted test pattern is at most a few percent, so there is a large difference in circuit switching activity between patterns filled with these two techniques. We do not use a compacted pattern set here, since the fill-rate of the compacted test sets is relatively high. Un-compacted test sets can magnify the difference between minimum transition and random fill techniques with a low fill-rate.

To compare the delay with high and low noise in a visual way, we plot the distribution of delay for the three benchmark circuits in Figure 32, Figure 33 and Figure 34. These figures show that random filling generally produces longer delays than minimum transition fill, due to the higher supply noise level. Minimum transition fill, however, produces longer delays than nominal delay, which is calculated without noise.

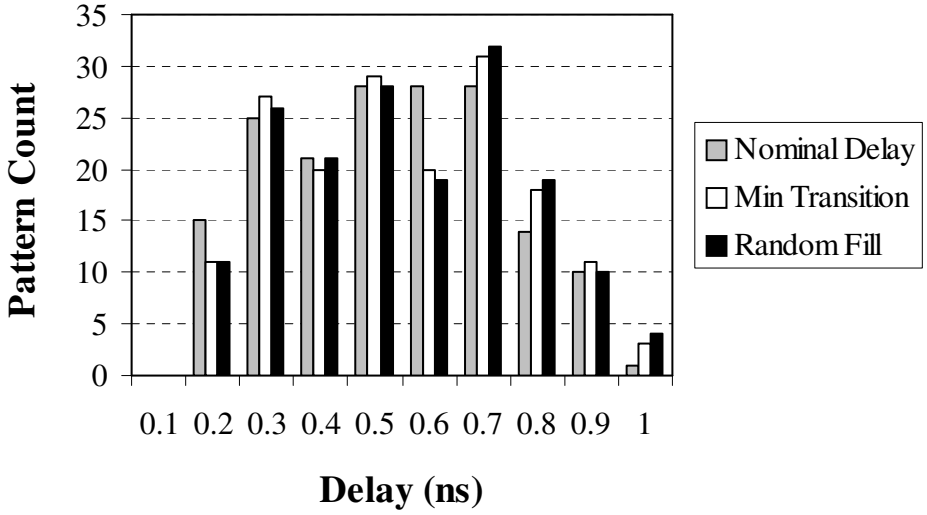

Figure 32. Delay histogram with minimum transition fill and random fill on s1488.

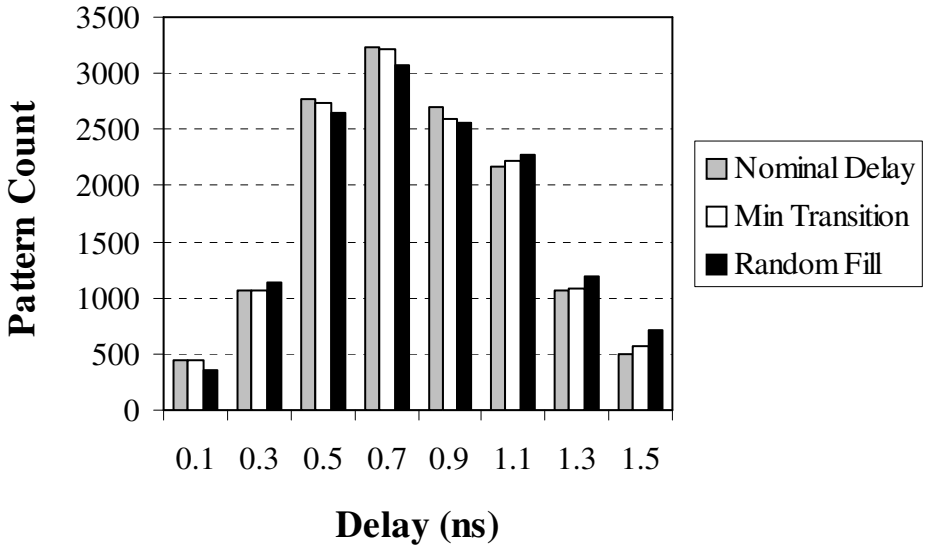

Figure 33. Delay histogram with minimum transition fill and random fill on s38417.

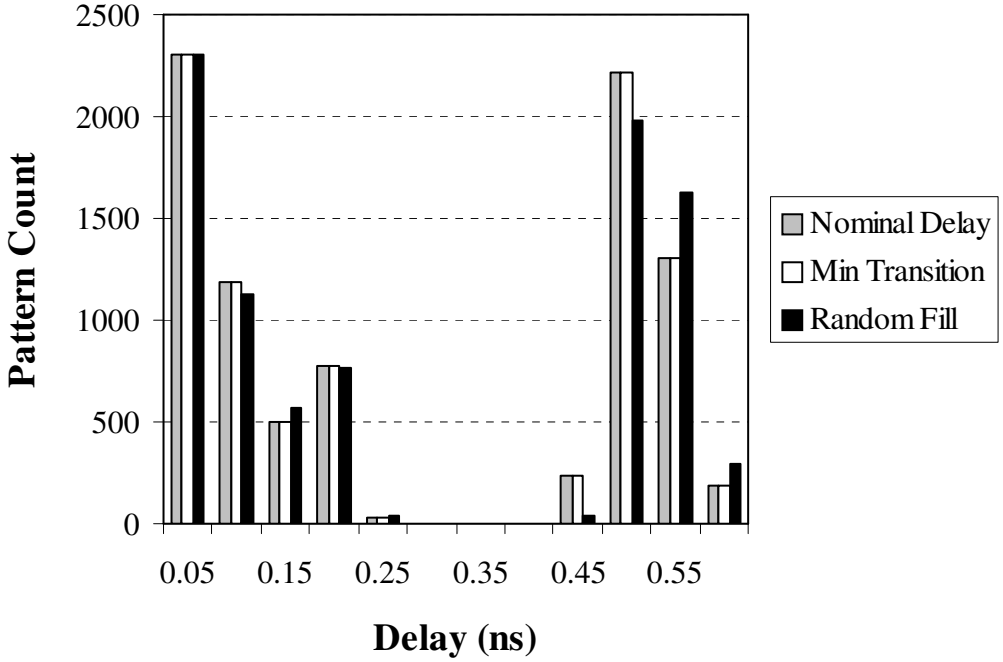

Figure 34. Delay histogram with minimum transition fill and random fill on s35932.

5.1.4 Transition Count Prediction Results for Dynamic Compaction

The transition count prediction algorithm designed for dynamic compaction was introduced in section 4. The experiments for transition count prediction were conducted using a different program. In our power supply noise analysis, zero-delay logic simulation is the step for further analysis. It tells us the initial and final value for each signal line when a delay test pattern is applied. Therefore, we are able to get the exact transition count of the circuit using logic simulation, excluding transitions due to glitches. The transition count prediction algorithm is applied to the circuit, which includes both PRE_COMPACTION_CIRCUIT_ANALYSIS, as introduced in Figure 20, and TRANSITION_COUNT_ESTIMATION, as introduced in Figure 22. The predicted transition count results are then compared with logic simulation results to see

how they correlate with each other.

The predicted transition count is the expectation of transition count given a series of signal lines with known values. An experiment design to validate the transition count prediction results should consists of the following steps:

1. Given a set of signal lines with known values, use our transition count prediction algorithm to find the expectation of transition count.
2. Generate a number of input patterns consistent with these known values.
3. Calculate transition count using logic simulation as each of the input patterns is applied to the circuit, and then compute the average transition count.
4. Compare the average transition count in step 3 with the expectation of transition count in step 1.

However, this experiment design is not easy to implement due to step 2. An ATPG-like tool is needed to generate input patterns that are consistent with given signal values. In addition to this, the goal of this algorithm, which computes the expectation of transition count, is to serve dynamic compaction such that it can give warnings on the compaction trials that may lead to high power supply noise. Therefore, the final target is not to estimate the accuracy of this expectation value, but to efficiently to give warnings on high noise during dynamic compaction.

Therefore, our experiments are designed as follows:

1. Given a test pattern, calculate transition count using logic simulation as this test pattern is applied to the circuit.
2. Randomly select a number of signal lines from the circuit and record their initial

and final value in logic simulation using this test pattern.

3. Given the values of these signal lines, apply the transition count prediction algorithm to calculate expectation of transition count.
4. Repeat steps 2 and 3 several times, and calculate the average of transition count expectation given values on different signal lines.
5. Compare the actual transition count from step 1 with the average value of transition count expectation in step 4.

This experiment design tries to simulate situations met in dynamic compaction. For each interim test pattern along with a series of signal lines with known values, our transition count prediction algorithm can be used to produce an expectation value of transition count. The ATPG and compaction algorithm determines which signal lines of the circuit have known values during dynamic compaction. The final complete test pattern after test generation may produce more or fewer transitions than the prediction, since the predicted transition count is an expectation. However, the more signal lines with known values, the closer the prediction becomes to the real transition count.

Experiments were performed on two ISCAS89 benchmark circuits, s38417 and s35932. For each circuit, two types of test sets were used. One was an uncompacted test set from CodGen, which usually has a low transition count, and the other was the compacted test set of the first one, which often leads to a noisy circuit. Patterns in both test sets were filled with minimum transition to minimize supply noise by test fill. Test patterns during dynamic compaction are expected to have intermediate noise level between these two test sets. Both static and robust delay test sets were used in the

experiments.

As we introduced in the algorithm discussion, one major source of error lies in the process of determining switching fan-in inputs. Here we apply both conservative and aggressive selection methods in the algorithm to get a low and a high expectation value of transition count, respectively, and show how they correlate with the real transition count from logic simulation. These two values are called “low prediction” and “high prediction” in the rest of this section. We want to shown in the experiments whether the actual transition count will fall between these two values.

In the experiments, for each test pattern under analysis, we first randomly selected 1% of the signal lines from the whole circuit and recorded their values during logic simulation. The transition count prediction algorithm was executed using these signal values. This procedure was repeated 5 times, and the average of the 5 prediction values was computed. The signal percentage was then increased to 5% and 10% to show how correlation improves as more signal values are known. For either uncompacted or compacted test sets, we selected the first 200 test patterns for experiments.

The correlations between high prediction and actual transition count on circuit s38417 using static uncompacted test patterns are shown in Figure 35, Figure 36 and Figure 37 with 1%, 5% and 10% known signals, respectively. As expected, the more signal values that are known, the better the correlation between prediction and actual switching count. However, even for 1% of known signals, as shown in Figure 35, the correlation is already as high as 0.97.

Since the uncompacted test patterns are expected to generate lower transition counts

than average, the high prediction should overestimate the transition count in most cases. We also find that the high prediction value increases as more signal values are known. This is because for a small number of signal values, not many transitions are known to exist in the circuit. Thus, some switching circuit input will be missed, since none of their fan-out transitions are known. This will be improved as more and more signal values are selected for analysis. Therefore, in Figure 35, the high prediction value underestimates the transition count. But in Figure 36, with 5% of known gate outputs, the high prediction can be safely taken as an upper bound, and the same in Figure 37.

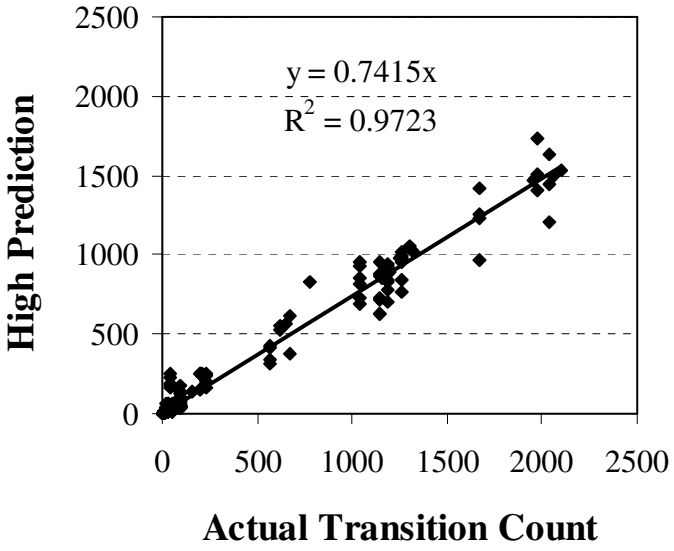

Figure 35. High prediction vs. actual transition count on circuit s38417 with 1% signal values, using static uncompacted test set.

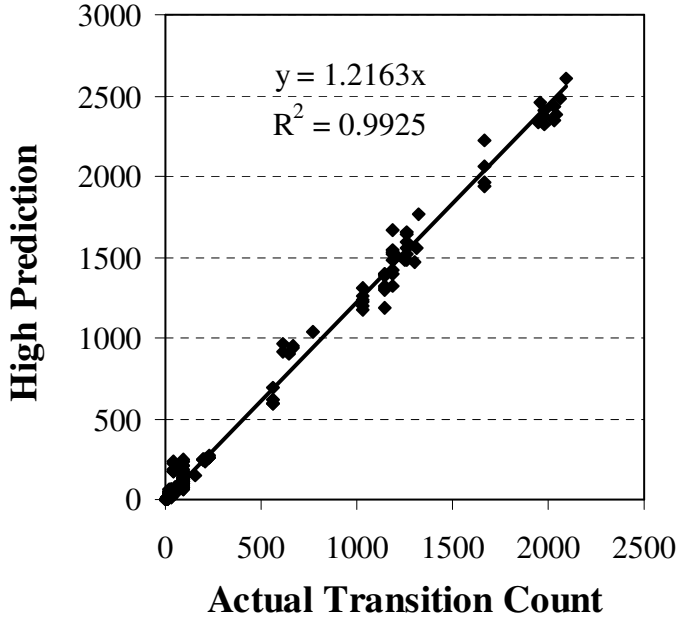

Figure 36. High prediction vs. actual transition count on circuit s38417 with 5% signal values, using static uncompacted test set.

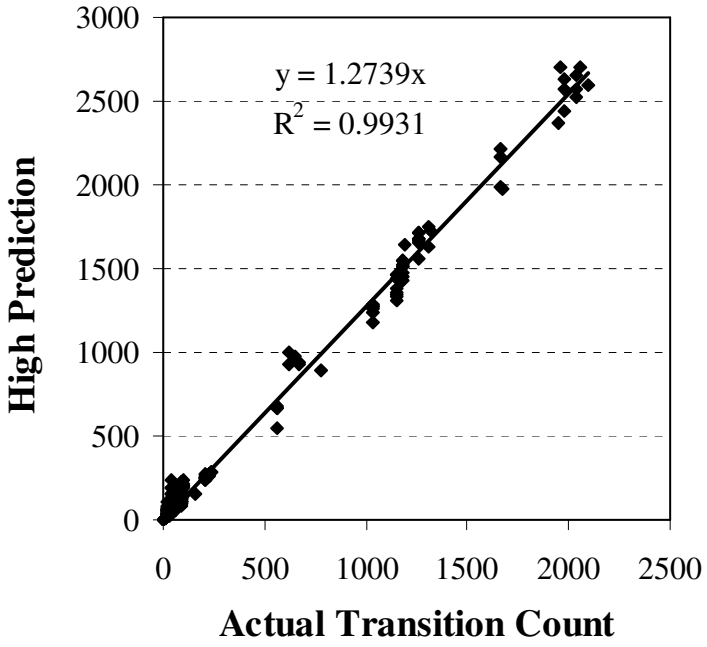

Figure 37. High prediction vs. actual transition count on circuit s38417 with 10% signal values, using static uncompacted test set.

The correlations between low prediction and actual transition count for circuit s38417 are shown in Figure 38, Figure 39 and Figure 40 with 1%, 5% and 10% known signals, respectively. The same static uncompacted delay test sets are used as in Figure 35, Figure 36 and Figure 37. Similar to the high prediction results, the more signal values that are known, the better the correlation between prediction and actual switching count. In addition, the correlation is already as high as 0.95 even with 1% of known signals. Considering the high correlation in high prediction results (shown in Figure 35, Figure 36 and Figure 37), our prediction algorithm has a very good correlation with actual transition count for low-noise test patterns.

However, the low prediction value cannot safely serve as a lower bound for low-noise test patterns. The low prediction with 1% known signals largely underestimates the transition count due to the same reason mentioned for Figure 35. Once more signal values are known, the low prediction tends to overestimate compared with actual transition count. This is because the low prediction is still an expectation of transition count, though using a conservative method in determining switching circuit inputs. In most cases, the uncompacted delay test patterns should generate fewer transition counts than the expectation value.

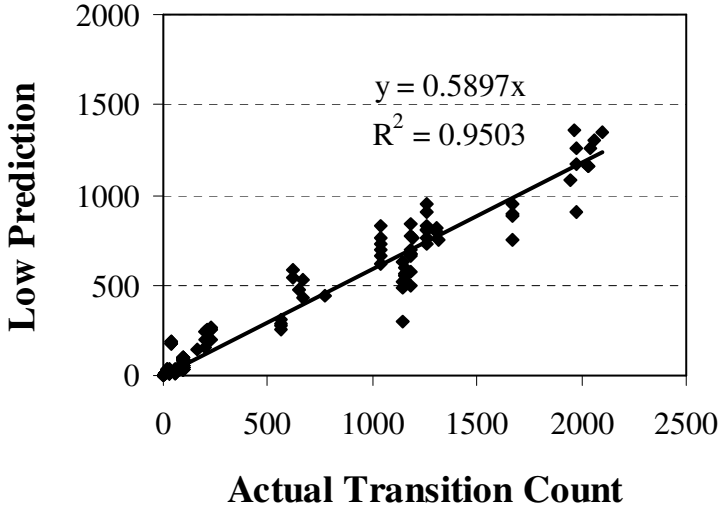

Figure 38. Low prediction vs. actual transition count on circuit s38417 with 1% signal values, using static uncompacted test set.

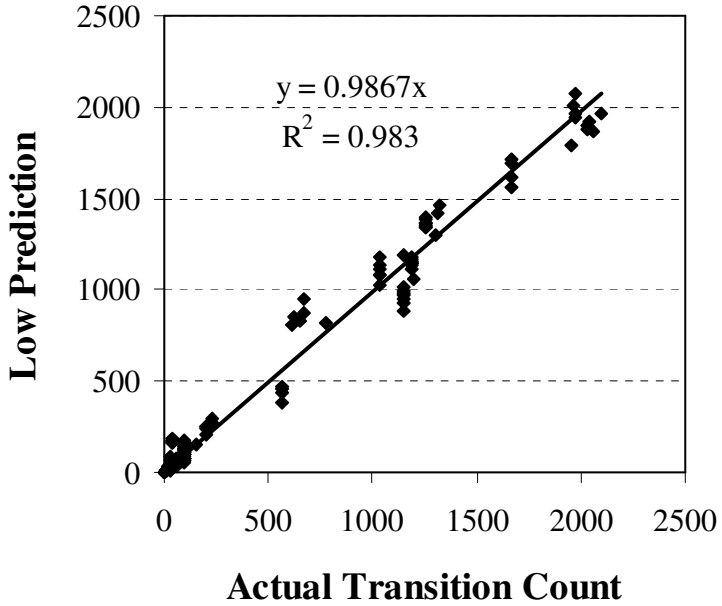

Figure 39. Low prediction vs. actual transition count on circuit s38417 with 5% signal values, using static uncompacted test set.

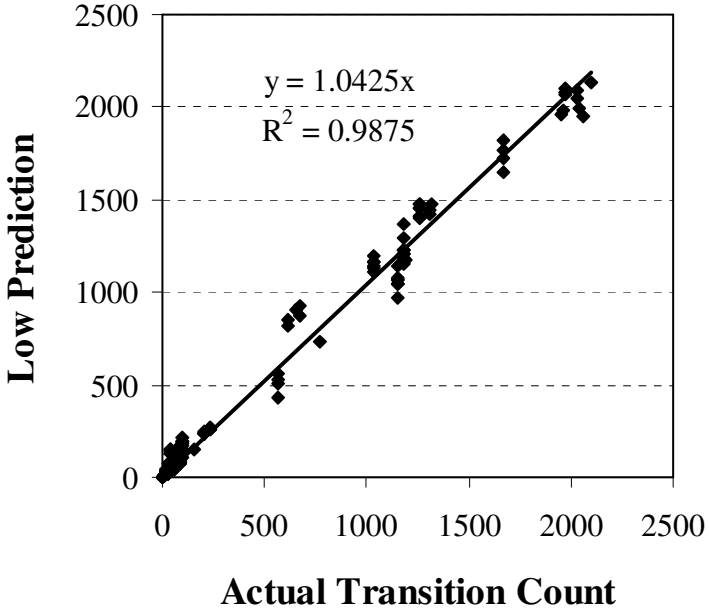

Figure 40. Low prediction vs. actual transition count on circuit s38417 with 10% signal values, using static uncompacted test set.

Now, we show both high prediction and low prediction results for circuit s38417 using the compacted test set, which is generated from the previous uncompacted test set using static compaction. The compacted test set is expected to generate more transitions than most patterns handled in dynamic compaction. Since the motivation of this algorithm is to give excess noise warnings during dynamic compaction, the results on high noise patterns are more important than low noise patterns. The results are shown in the next 6 figures, from Figure 41 to Figure 46.

The correlation for either high prediction or low prediction results are not as good as low noise patterns. Note that for 5% and 10% known signals , correlations with both zero y-intercept and non-zero y-intercept are shown. For high prediction results with 10% known signals, the correlation is 0.68 with zero y-intercept, and 0.86 with non-zero

y-intercept. The low prediction correlations are similar. With 10% known signals, the low prediction correlation is 0.74 with zero y-intercept, and 0.82 with non-zero y-intercept. The correlation becomes better as more signal values are known, as expected. Therefore, for high noise test patterns, we need more signal values to achieve comparable accuracy as achieved for low noise test patterns.

Neglecting the high prediction with 1% known signals, the high prediction result is larger than the actual transition count for most test patterns when 5% or more of the signal values are known. This means our high prediction results can safely serve as an upper bound of transition count even for high noise test patterns.

Low prediction results, however, are similar to the results for low noise test patterns. With 10% known signals, our low prediction results slightly overestimate compared with the actual transition count. Therefore, low prediction results cannot safely serve as lower bound for the actual transition count. The main reason is that in determining switching circuit inputs using conservative method, we use a simple first-come-first-serve strategy instead of solving a min-cover problem. Therefore, a better approach is desired for low prediction in future work.

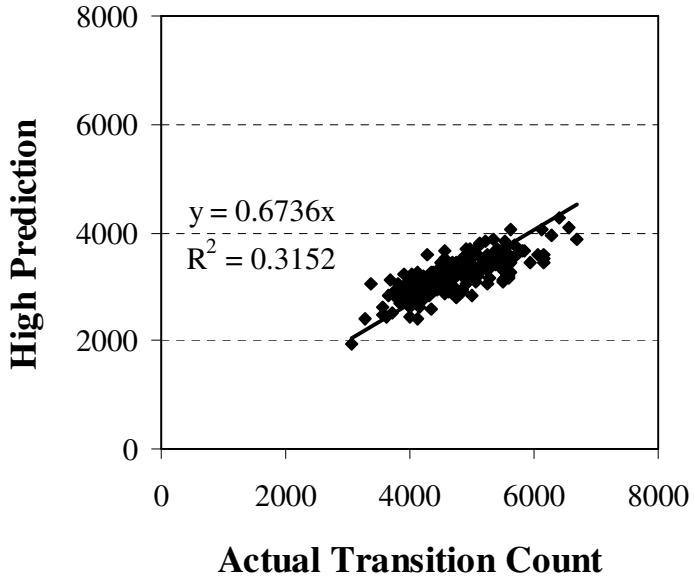

Figure 41. High prediction vs. actual transition count on circuit s38417 with 1% signal values, using static compacted test set.

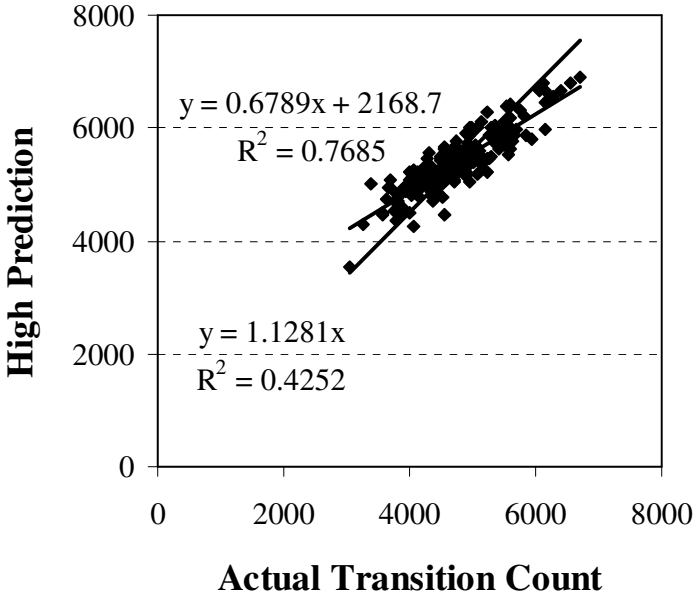

Figure 42. High prediction vs. actual transition count on circuit s38417 with 5% signal values, using static compacted test set.

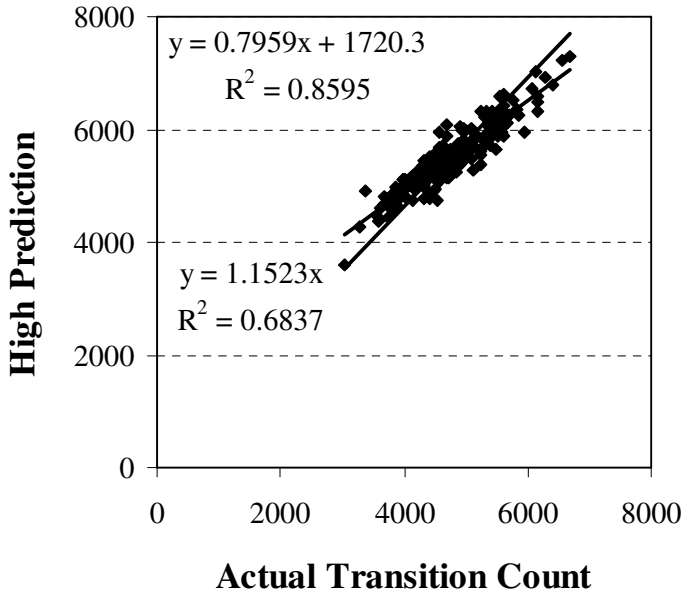

Figure 43. High prediction vs. actual transition count on circuit s38417 with 10% signal values, using static compacted test set.

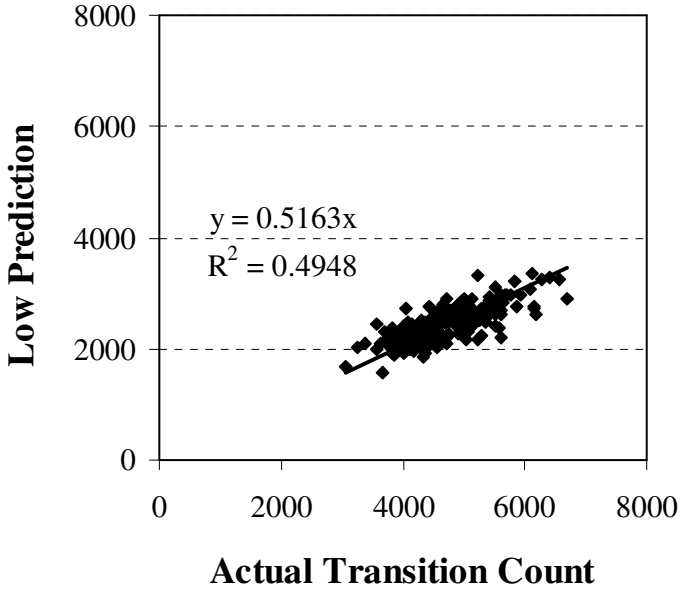

Figure 44. Low prediction vs. actual transition count on circuit s38417 with 1% signal values, using static compacted test set.

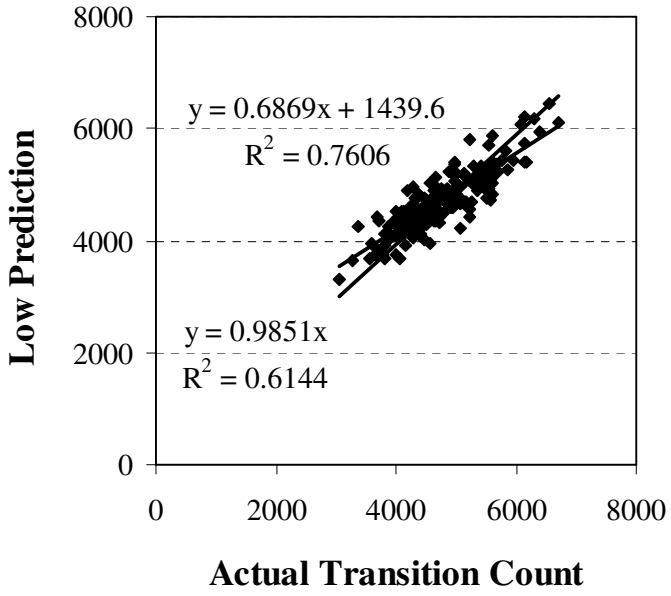

Figure 45. Low prediction vs. actual transition count on circuit s38417 with 5% signal values, using static compacted test set.

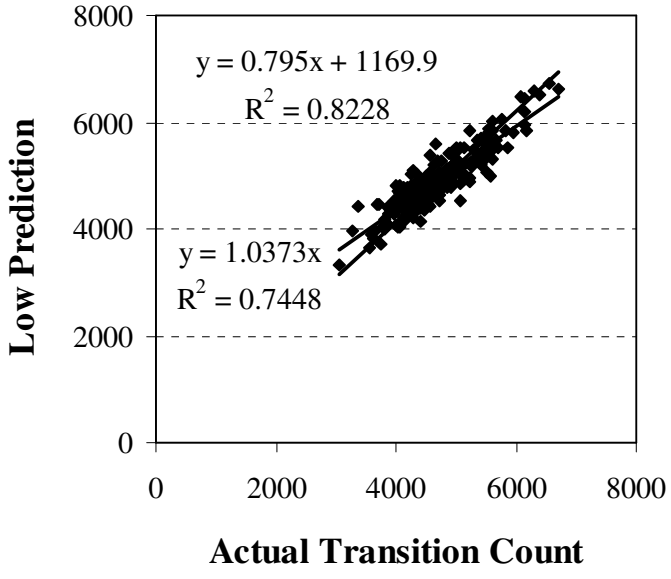

Figure 46. Low prediction vs. actual transition count on circuit s38417 with 10% signal values, using static compacted test set.

Conclusions can be drawn based on the s38417 experimental data on both low noise test patterns and high noise test patterns introduced in this section.

First, the more signals we know, the better prediction we get. No convincing prediction can be computed if too few signal values are known. On the other hand, once we have a certain number of signal values, the prediction efficiency and accuracy will not improve much as more signal values are known.

Second, our high prediction results work well as an upper bound for both low noise and high noise test patterns. Our low prediction results, in contrast, usually slightly underestimate transition count. Better approaches in determining switching circuit inputs are desired for low prediction calculation.

The experiments using robust delay test sets, and the experiments on circuit s35932, all show similar results to the experiments described above.

5.2 Experiments on an Industrial Design

The second set of experiments is based on Noise Model II. It is performed on an wire bond industrial design from NXP Semiconductors. The experiments focus on validation of Noise Model II with silicon data. Compaction experiments have not been performed due to limited resources.

The circuit under test (CUT) is a DSP-like core of a test chip in a 160-pin quad flat pack (QFP). The core is fabricated in a 130 nm technology with a standard cell library. No dynamic logic is used and the circuit contains more than 1 million transistors. The nominal supply voltage is 1.2 V. The same device was used in a study of fine delay fault

detection by Kruseman et al. [71].

A power supply noise analysis tool, based on Noise Model II, was developed in C++ and run on a 2.3 GHz Pentium 4 system. This design has only one effective region since the calculated RC time constant of the whole chip area is less than the clock cycle time. Off-chip current is neglected in the experiments.

Delay measurements are taken directly from the tester. A step size of 25 ps is employed in the measurements. This will possibly lead to discretization in the delay data. Also the data measurement is repeated on silicon to make sure it is consistent.

The launch-on-capture path delay patterns generated by an internal ATPG tool leave all the unassigned pattern bits as don't care. The target paths are all statically sensitized such that all side inputs of the path are restricted to be static non-controlling. This ensures transitions propagate on the target path in our experiments.

5.2.1 Test Fill Results

The filling strategy we adopt in this work is to randomly set these "don't care" bits to 1 with a specified probability. The term 1-filling rate is defined as the probability of assigning a "don't care" bit to 1 during pattern filling. For each unfilled pattern, we vary this 1-filling rate and generate a number of patterns at each rate and eventually create a pattern set. All test patterns in such a pattern set target one path, which is also the target path of the unfilled pattern.

Four unfilled test patterns, each targeting one path, were used in our experiments. Each test pattern was filled ten times randomly at each 1-filling rate, from 0% to 100% in 10% steps. Hence, a set of 92 filled patterns was generated per unfilled test pattern,

and we get four pattern sets in total. They are called “set 1”, “set 2”, “set 3” and “set 4” in the rest of the section. Set 2 targets the longest path among the four paths. Target paths for set 3 and set 4 are also long paths compared with most paths on the circuit. Set 1 targets a relatively short path.

For patterns in each of these test sets, we expect to have different noise impact on path delay due to switching activity produced by different don’t care bit assignments. Figure 47 plots the silicon delay measurements versus 1-filling rate for set 2, which targets the longest path among the four paths considered in the experiments. The result shows that the delay variation between high noise and low noise is as much as 0.85 ns, which is 15% of the path delay, assuming the smallest delay value measured is the nominal delay.

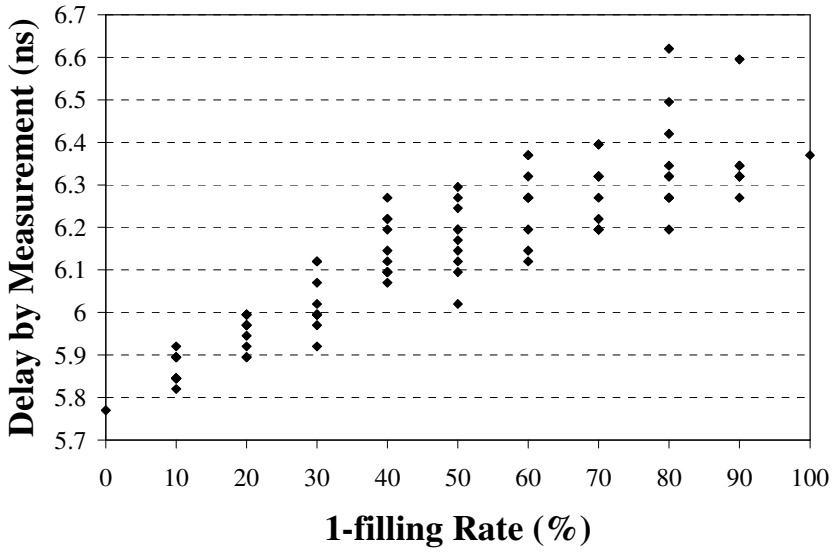

Figure 47. Measured path delay vs. 1-filling rate for set 2.

Figure 47 also shows that for the target path, a higher 1-filling rate generally produces

a longer delay. In Figure 47, the maximum average delay appears at 90% 1-filling rate. This phenomenon is due to the particular characteristics of the circuit function and design. Test patterns with a higher 1-filling rate are more likely to generate more switching activity. Note that we use launch-on-capture patterns, so 100% 1-filling can still result in any activity level between 0 and 100%. In contrast, 100% 1-filling of launch-on-shift patterns would produce an activity close to 0%. One circuit characteristic that causes this skewed behavior for 1-filling is a heavy usage of AND/NAND gates in the first stages of the paths. If either or both inputs change state, a transition occurs. This is in contrast to 0–filling, in which both inputs need to change state to create a transition.

A simple metric to investigate the activity is to count the number of signal transitions. Figure 48 plots the 1-filling rate versus the total number of simulated signal transitions, using the same test set as in Figure 47. The switching count includes the rising and falling transitions on all signal nets. Glitches during the propagation phase are also considered as full transitions. We find in Figure 48 that higher 1-filling rates cause more activity.

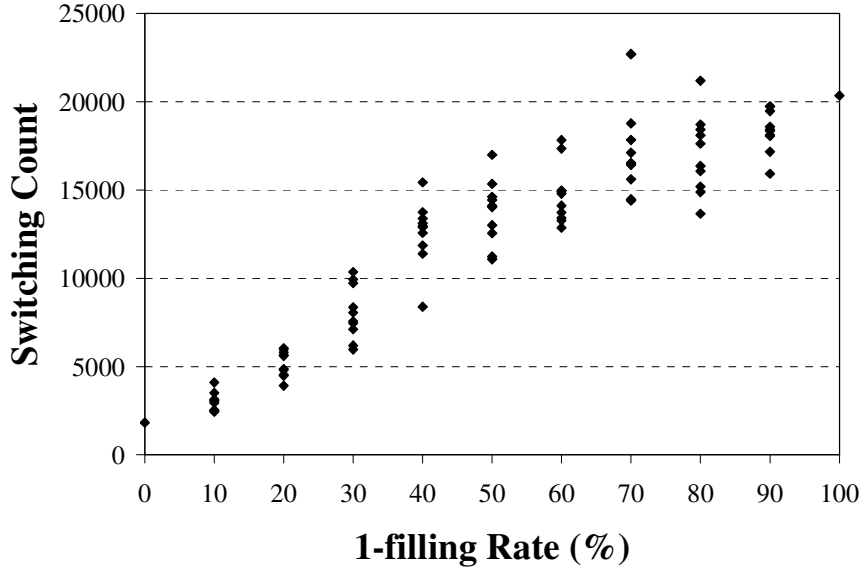

Figure 48. Switching count vs. 1-filling rate for set 2.

The delay variation data versus 1-filling rate for the other three test sets is plotted in Figure 49, Figure 51 and Figure 53. For set 3 and set 4, the delay variation is not as large as for set 2, but is still significant. Also the path delay for these two sets increases with 1-filling rate, showing the same trend as set 2. The exception is set 1, which targets a relatively short path. For this path, we only observe an increase in delay for 1-filling rates of 70% and more. The delay variation is also much smaller than other test sets. One explanation is that the propagation on the short path ends before many transitions on average length paths occur. As a consequence, the voltage drop impact on delay for this path is smaller than for other paths.

The transition count data versus 1-filling rate for the other three test sets is plotted in Figure 50, Figure 52 and Figure 54. All three test sets show that switching count

variation show similar trend as delay variation as 1-filling rate increases, same as our observation for set 2.

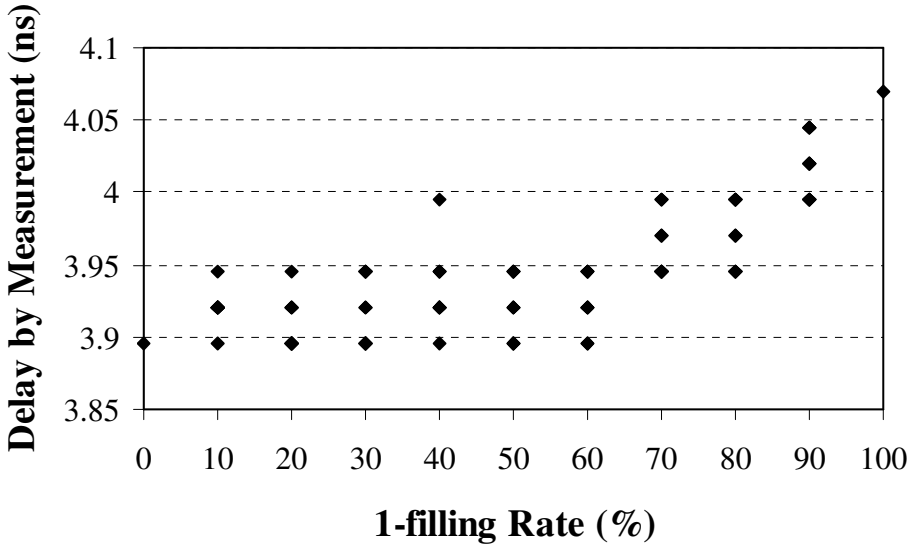

Figure 49. Measured path delay vs. 1-filling rate for set 1.

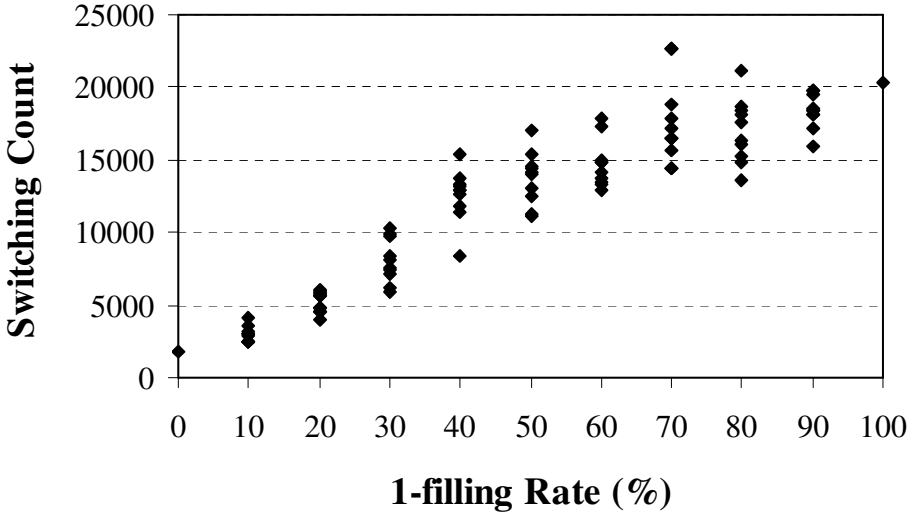

Figure 50. Switching count vs. 1-filling rate for set 1.

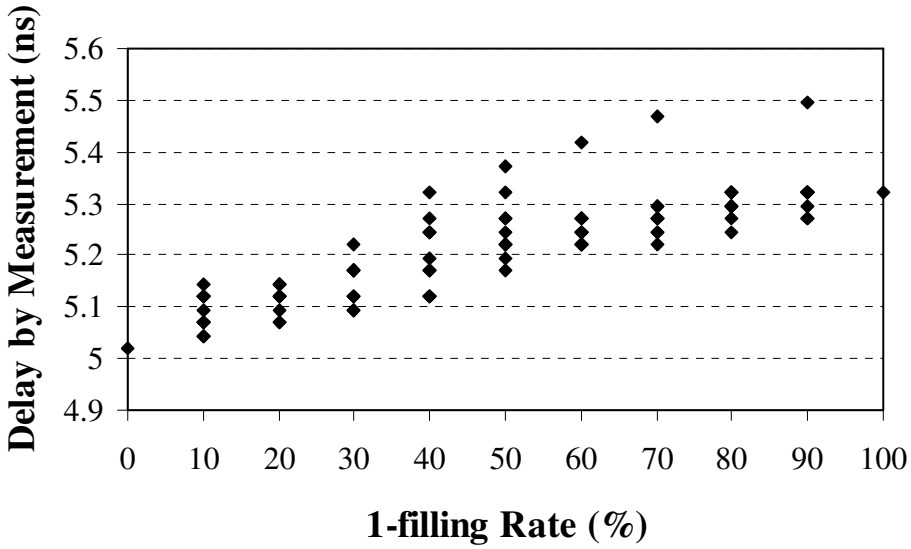

Figure 51. Measured path delay vs. 1-filling rate for set 3.

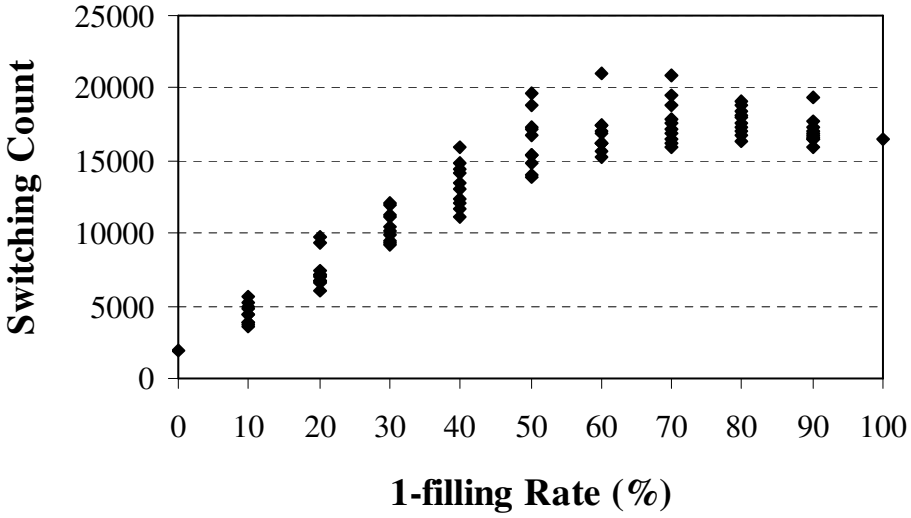

Figure 52. Switching count vs. 1-filling rate for set 3.

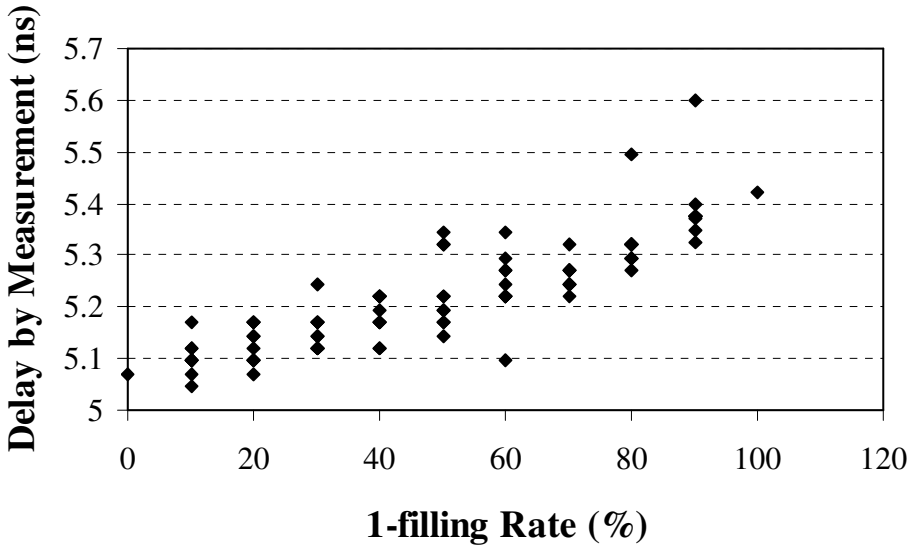

Figure 53. Measured path delay vs. 1-filling rate for set 4.

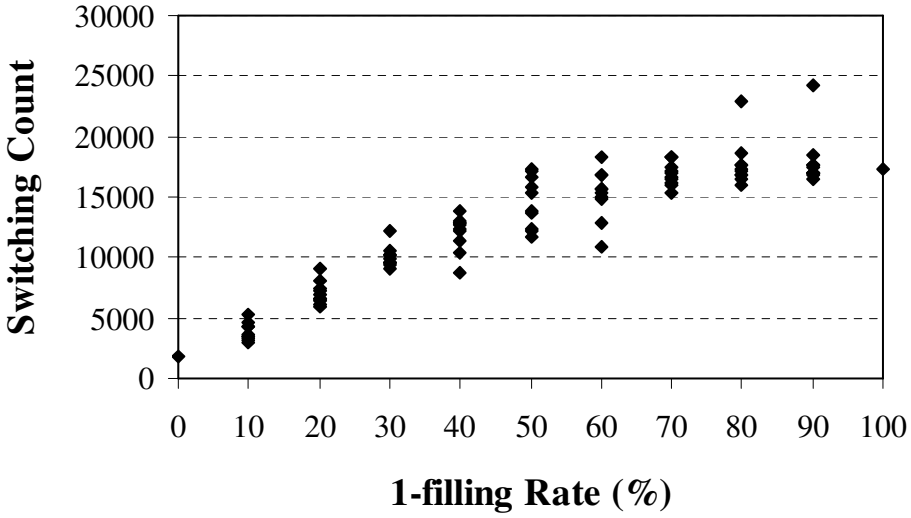

Figure 54. Switching count vs. 1-filling rate for set 4.

5.2.2 Validation of Noise Model II

Although Figure 47 and Figure 48 confirm the dependence between activity and

delay, it does not quantify it. For this we use our timing analysis tool, which is based on Noise Model II. Figure 55 shows the delay measured by the tester versus the delay from our timing analysis tool for set 2, the same test set as in Figure 47 and Figure 48. The correlation is 0.83 with an intercept that is non-zero. A zero-intercept was not expected since there can be a variety of errors between simulation and measurement. In this research we are not interested in an absolute agreement, only a relative one. The offset is especially clear in Figure 56, which shows the measurements for each pattern, simulated nominal delay, and simulated noise-induced delay increase. Both the simulated nominal delay value and the simulated delay increase vs. the 1-filling rate are lower than the measurements. This is due to shortcomings in the delay model characterization. The correlation, however, shows that extra delay measured on the tester can be well explained by the impact of power supply noise.

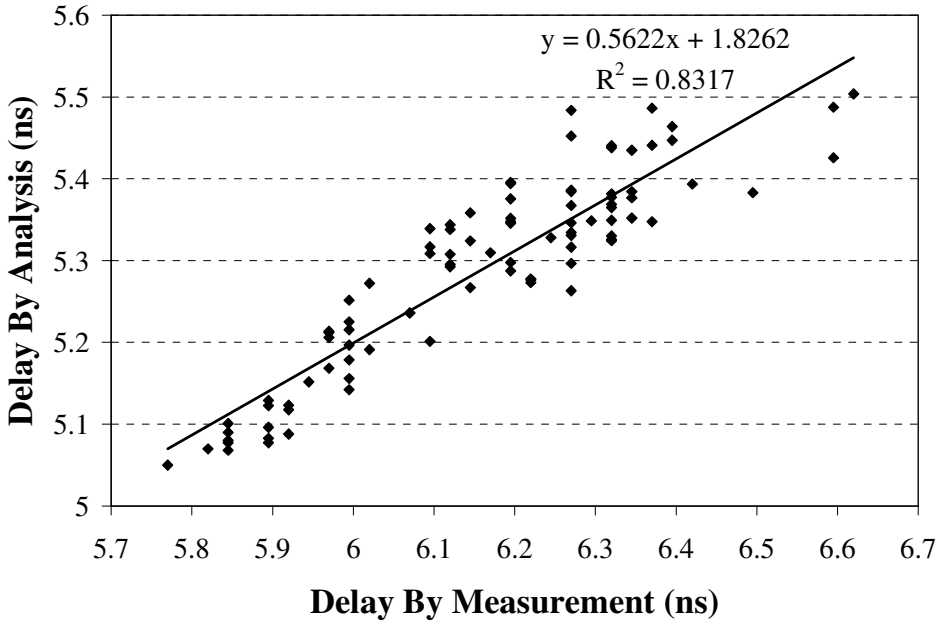

Figure 55. Tester delay vs. timing analysis delay for set 2.

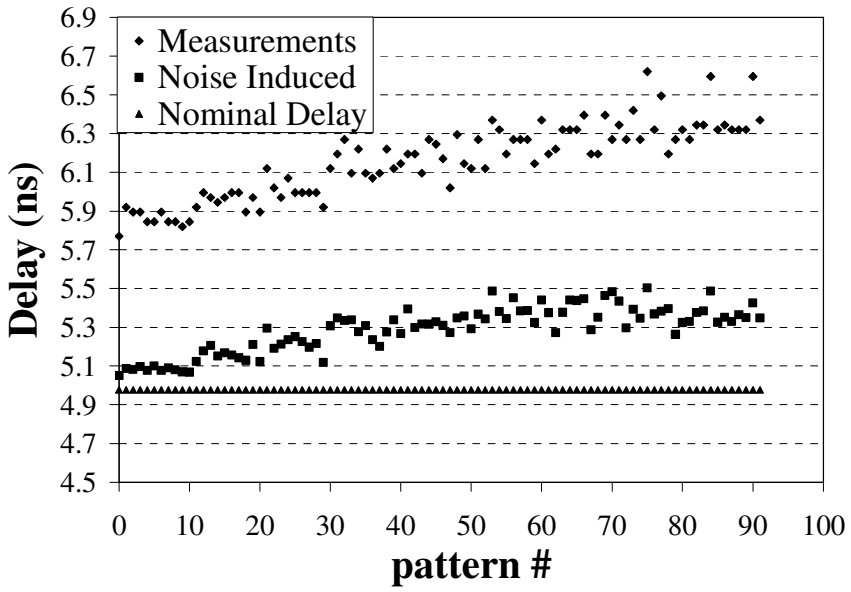

Figure 56. Nominal and noise induced delay by analysis, and measured delay be tester for the same test set.

The correlation results for set 3 and set 4 are shown in Figure 57 and Figure 58, respectively. These two sets target long paths in the circuit, though shorter than the target path of set 2. Their correlation results are also similar to test set 2.

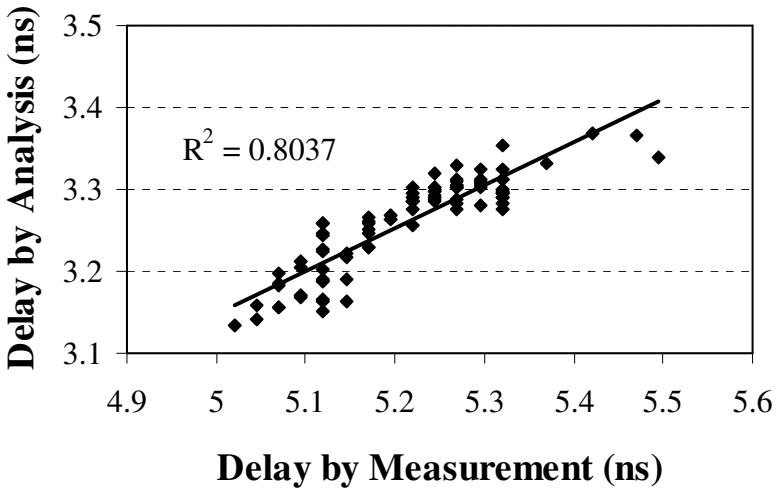

Figure 57. Tester delay vs. timing analysis delay for set 3.

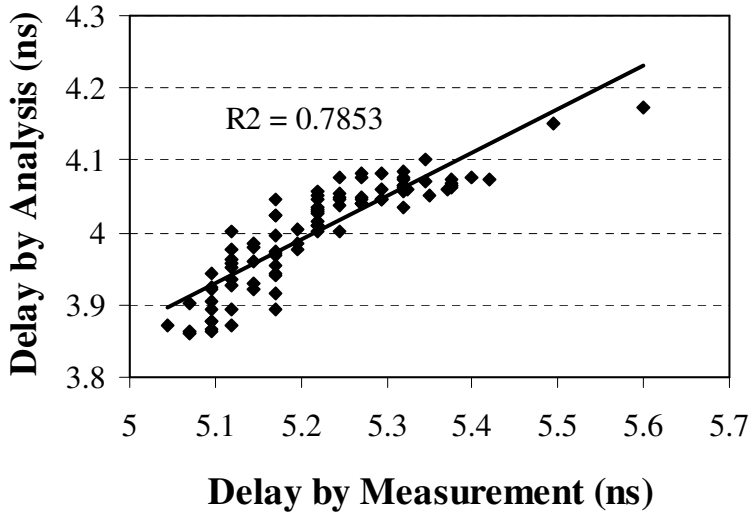

Figure 58. Tester delay vs. timing analysis delay for set 4.

Experiments on set 1, which targets a relatively short path, do not show good correlation between delay and noise in Figure 59. This is not surprising, since voltage drop impact on delay for short paths is also smaller, as shown in Figure 49.

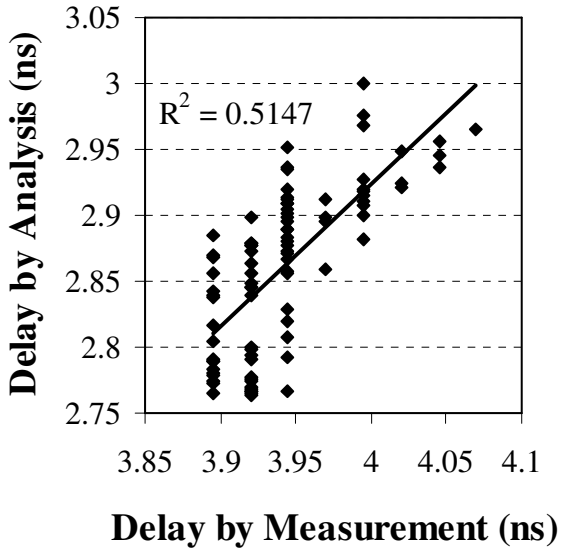

Figure 59. Tester delay vs. timing analysis delay for set 1.

5.2.3 Operating Conditions

Different operating conditions may have a significant impact on delay. The main factors in operation conditions are nominal voltage, temperature and process parameters. When operating conditions change, delay model parameters should change as well.

In our experiments, we take set 2 again, and show delay variation for each test pattern under three operating conditions. The Nominal operating conditions mean normal voltage, room temperature and medium process variation. The Worst Case operating conditions include low voltage, high temperature and large process variation. The Best Case operating conditions include high voltage, low temperature and small process variation. The simulation results using different delay model parameters under different operating conditions are shown in Figure 60. We can see that the delay difference due to

operating conditions is significant. This also helps to explain the large offset shown in Figure 56.

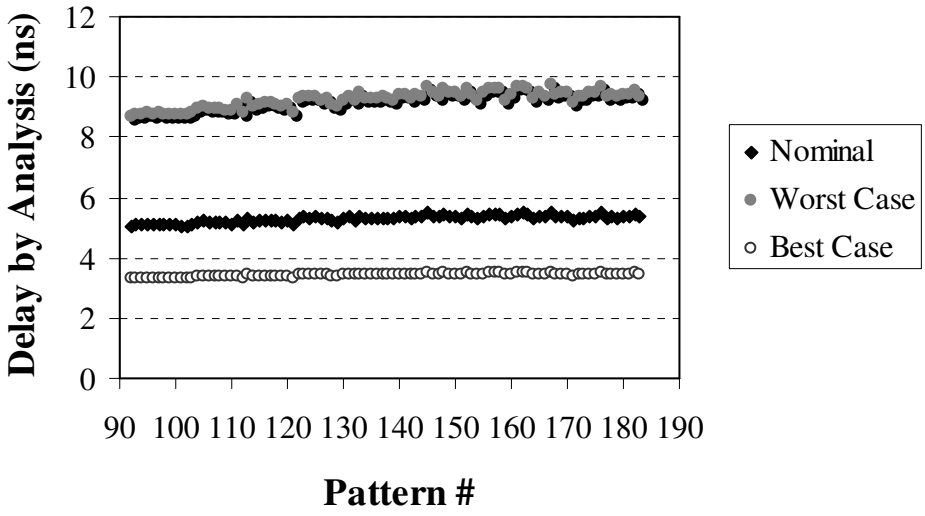

Figure 60. Delay variation under different operating conditions for set 2.

5.3 Comparison of the Noise Models

As we compare the validation results for these two models, it might appear that the correlation for Noise Model I is significantly better than Noise Model II. However, the experiments on Noise Model I were based on circuit simulation, while the experiments for Noise Model II were based on packaged silicon on the tester. In addition, ISCAS benchmark circuits are much smaller than the industrial design, and the cell library delay model is not as sensitive to supply noise as the delay model for the industry standard library. Taking these factors into account, the performance of the two noise models can be considered comparable.

6. SUMMARY AND FUTURE WORK

We have addressed the delay test overkill problem due to excessive noise-induced delay produced in delay test generation. The excessive noise comes from test compaction and test fill. None of the previous work addressed this problem.

We have proposed an approach to analyze pattern-dependent noise-induced delay during delay test. Two low-cost noise models have been proposed to address array bond and wire bond power supply networks, and experimentally validated using ISCAS89 benchmark circuits as well as an industrial design. We found that Noise Model I works well for array bond chips and Noise Model II works well for wire bond chips. Because Noise Model II better characterizes local voltage variation, it has the potential to be more accurate. In future work, Noise Model II will be modified so that its region analysis can handle array bond chips, and include off-chip current. This will allow Noise Model II to take the place of Noise Model I.

A noise constraint static compaction tool was developed based on a greedy compaction algorithm and our supply noise analysis approach. Experiments were performed on ISCAS89 benchmark circuits. Results show that compacted delay test patterns generated by our compaction tool can meet a moderate noise or delay constraint with only a small increase in compacted test set size.

A transition count prediction algorithm was proposed and implemented to estimate average switching activity based on partial information on circuit signal values. Experimental data from ISCAS89 benchmark circuits shows it can efficiently predict the upper bound of circuit transition count with limited signal value information. In future

work, we need to improve this algorithm so that it can more accurately predict the lower bound of transition count as well. We also want to make it a layout-aware approach to more efficiently control switching activity and supply noise during dynamic compaction. This algorithm should be integrated to a dynamic compaction tool in future work to eliminate excessive supply noise.

Traditionally, don't care bits are randomly filled to increase fortuitous defect detection. In our work, a test fill tool with supply noise analysis was developed. We showed by experiments that the filling strategy can have a significant impact on switching activity, power supply noise and delay. Therefore, we need to take noise into consideration once test fill is applied to delay test patterns.

Many circuits include embedded memory arrays that are treated as black boxes during ATPG. Prior research [97 has shown that testing the longest paths through these arrays is necessary to accurately test chip speed. If a behavioral model of the arrays is supplied, the ATPG can test paths through them. However, considering noise during these tests require a low-noise model for the arrays.

REFERENCES

[1] ITRS, "International Technology Roadmap for Semiconductors (ITRS)," [Online]. Available: http://www.itrs.net

[2] S. Pant, D. Blaauw, V. Zolotov, S. Sundareswaran and R. Panda, "Vectorless Analysis of Supply Noise Induced Delay Variation," in *Proc. ACM/IEEE Int. Conf. Comput.-Aided Design*, 2003, pp. 184-191.

[3] C. Tirumurti, S. Kundu, S. Sur-Kolay and Y.-S. Chang, "A Modeling Approach for Addressing Power Supply Switching Noise Related Failures of Integrated Circuits," in *Proc. Design Autom. Test Eur. Conf. Exhibition*, vol. 2, 2004, pp. 1078-1083.

[4] Leach, W.M., Jr., "Fundamentals of Low-noise Analog Circuit Design," *Proc. IEEE*, vol. 82, no. 10, pp. 515-1538, Oct. 1994.

[5] P. R. Gray and R. G. Meyer, *Analysis and Design of Analog Integrated Circuits*, John Wiley and Sons, 1984, Chapter 11.

[6] P. Larsson and C. Svensson, "Noise in Digital Dynamic CMOS Circuits," *IEEE J. Solid-State Circuits*, vol.29, no. 6, pp. 655-661, Jun. 1994.

[7] K. L. Shepard and V. Narayanan, "Noise in Deep Submicron Digital Design," in *Proc. IEEE Int. Conf. Comput.- Aided Design*, 1996, pp. 524-531.

[8] L. D. Smith, H. R. Farmer, M. Kunesh, M. A. Massetti, D. Willmott, R. Hedman, R. Richetta and T. J. Schmerbeck, "A CMOS-based Analog Standard Cell Product Family," *IEEE J. Solid-State Circuits*, vol. 24, no. 2, pp. 370-379, Apr. 1989.

[9] V. Zolotov, D. Blaauw, S. Sirichotiyakul, M. Becer, C. Oh, R. Panda, A. Grinshpon, R. Levy, “Noise Propagation and Failure Criteria for VLSI Designs,” in *Proc. IEEE Int. conf. Comput.-Aided Design*, 2002, pp. 587-594.

[10] K. L. Shepard, “Design methodologies for noise in digital integrated circuits,” in *Proc. Design Autom. Conf.*, 1998, pp. 94-99.

[11] H. H. Chen and D. D. Ling, “Power Supply Noise Analysis Methodology for Deep Submicron VLSI Chip Design,” in *Proc. Design Autom. Conf.*, 1997, pp. 638-643.

[12] Y.-M. Jiang and K.-T. Cheng, “Analysis of Performance Impact Caused by Power Supply Noise in Deep Submicron Devices,” in *Proc. Design Autom. Conf.*, 1999, pp. 760-765.

[13] S. Mukhopadhyay, A. Raychowdhury and K. Roy, “Accurate Estimation of Total Leakage Current in Scaled CMOS Logic Circuits Based on Compact Current Modeling,” in *Proc. Desing Autom. Conf.*, 2003, pp 169-174.

[14] A. Agarwal and K. Roy, “A Noise Tolerant Cache Design to Reduce Gate and Sub-threshold Leakage in the Nanometer Regime,” in *Proc. Int. Symp. Low-Power Electron. Design*, 2003, pp. 18-21.

[15] S. Mukhopadhyay, C. Neau, R. T. Cakici, A. Agarwal, C. H. Kim and K. Roy, "Gate Leakage Reduction for Scaled Devices Using Transistor Stacking," *IEEE Trans.Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 4, pp. 716-730, Aug. 2003.

[16] A. Abdollahi, F. Fallah and M. Pedram, "Runtime Mechanisms for Leakage Current Reduction in CMOS VLSI Circuits," in *Proc. Int. Symp. Low-Power Electron. Design*, 2002, pp. 213-218.

[17] A. R. Conn, R. A. Haring and C. Visweswariah, "Noise Considerations in Circuit Optimization," in *Proc. ACM/IEEE Int. Conf. Comput.-Aided Design*, 1998, pp. 220-227.

[18] K. Heragu, M. Sharma, R. Kundu and R. D. Blanton, "Test Vector Generation for Charge Sharing Failures in Dynamic Logic," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 21, no. 12, pp. 1502-1508, Dec. 2002.

[19] S. Kundu, S. T. Zachariah, Y. S. Chang and C. Tirumurti, "On Modeling Crosstalk Faults," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 24, no. 12, pp.1909-1915, Dec. 2005.

[20] W. Y. Chen, S. K. Gupta, and M. A. Breuer, "Test Generation in VLSI Circuits for Crosstalk Noise,"in *Proc. IEEE Int. Test Conf.*, 1998, pp. 641-650.

[21] J. Zhang and E. G. Friedman, "Crosstalk Noise Model for Shielded Interconnects in VLSI-based Circuits," in *Proc. IEEE Int. System.-on-Chip Conf.*, 2003, pp. 243-244.

[22] A. Sinha, S. K. Gupta and M. A. Breuer, "Validation and Test Issues Related to Noise Induced by Parasitic Inductances of VLSI Interconnects," *IEEE Trans. Advanced Packaging*, vol. 25, no. 3, pp. 329-339, Aug. 2002.

[23] 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," in *Proc. IEEE Int. Test Conf.*, 2003, pp. 1098-1104.

[24] R. Ahmadi and F. N. Najm, "Timing Analysis in Presence of Power Supply and Ground Voltage Variations,", in *Proc. ACM/IEEE Int. Conf. Comput.-Aided Design*, 2003, pp. 176-183.

[25] P. Larsson, "Power Supply Noise in Future IC's: A Cristal Ball Reading," in *Proc. IEEE Custom Integr. Circuits Conf.*, 1999, pp. 467-474.

[26] N. Weste and K. Eshragian, *Principles of CMOS VLSI Design*, Addison-Wesley, 1990.

[27] D. Draper, M. Crowley, J. Holst, G. Favor, A. Schoy, J. Trull, A. Ben-Meir, R. Khanna, D. Wendell, R. Krishna, J. Nolan, D. Mallick, H. Partovi, M. Roberts, M. Johnson and T. Lee, "Circuit Techniques in a 266-MHz MMX-Enabled Processor," *IEEE J. Solid-State Circuits*, vol. 32, no. 11, pp. 1650-1664, Nov. 1997.

[28] C. F. Webb, C. J. Anderson, L. Sigal, K. L. Shepard, J. S. Liptay, J. D. Warnock, B. Curran, B. M. Krumm, M. D. Mayo, P. J. Camporese, E. M. Schwarz, M. S. Farrell, P. J. Restle, R. M. Averill, T. J. Slegel, W. V. Houtt, Y. H. Chan, B. Wile, T. N. Nguyen, P. G. Emma, D. K. Beece, C. T. Chuang and C. Price, "A 400 MHz S/390 Microprocessor," *IEEE J. Solid-State Circuits*, vol. 32, no. 11, pp. 1665-1675, Nov. 1997.

[29] J. Schutz, R. Wallace, "A 450MHz IA32 P6 Family Microprocessor," in *Proc. Int. Solid-State Circuits Conf.*, 1998, pp. 236-237.

[30] S. Zhao, K. Roy and K. K. Cheng, "Decoupling capacitance allocation and its application to power-supply noise-aware floorplanning," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 21, no. 1, pp. 81-92, Jan. 2002.

[31] H. M. Chen, L. D. Huang, I. M. Liu, M. Lai and D. F. Wong, "Floorplanning with Power Supply Noise Avoidance," in *Proc. Design Autom. Conf. Asia and South Pacific (ASP-DAC)*, 2003, pp. 427-430.

[32] S. A. Moghaddam, N. Masoumi and C. Lucas, "A Stochastic Power-supply Noise Reduction Technique Using Max-flow Algorithm and Decoupling Capacitance," in *Proc. Int. Workshop System-on-Chip Real-Time Applications*, 2005. pp. 265-269.

[33] J. Gu, R. Harjani and C. Kim, "Distributed Active Decoupling Capacitors for On-Chip Supply Noise Cancellation in Digital VLSI Circuits," in *Proc. Symp. Very Large Scale Integr. (VLSI) Circuits, Digest of Technical Papers*, 2006, pp. 216-217.

[34] G. Keskin, X. Li and L. Pileggi, "Active On-Die Suppression of Power Supply Noise," in *Proc. IEEE Custom Integr. Circuits Conf.*, 2006, pp. 813-816.

[35] E. Alon, V. Stojanovic and M. A. Horowitz, "Circuits and Techniques for High-Resolution Measurement of On-chip Power Supply Noise," *IEEE J. Solid-State Circuits*, vol. 40, no. 4, pp. 820-828, Apr. 2005.

[36] A. Muhtaroglu, G. Taylor and T. Rahai-Arabi, "On-die Droop Detector for Analog Sensing of Power Supply Noise," *IEEE J. Solid-State Circuits*, vol. 39, Issue 4, pp. 651-660, Apr. 2004.

[37] C. Metra, L. Schiano and M. Favalli, "Concurrent detection of power supply noise," *IEEE Trans. Reliability*, vol. 52, no. 4, pp. 469-475, Dec. 2003.

[38] J. R. Vazquez and J. P. de Gyvez, "Power Supply Noise Monitor for Signal Integrity Faults," in *Proc. Design Autom. Test Eur. Conf. Exhibition*, vol. 2, 2004, pp.1406-1407.

[39] W. T. Eisenmann, "Fast Transient Power and Noise Estimation For VLSI Circuits," in *Proc. ACM/IEEE Int. Conf. Comput.-Aided Design*, 1994, pp. 252-257.

[40] Y. S. Cheng, S. K. Gupta and M. A. Breuer, "Analysis of Ground Bounce in Deep Sub-Micron Circuits," in *Proc. VLSI Test Symp.*, 1997, pp. 110-116.

[41] R. Panda, D. Blaauw, R. Chaudhry, V. Zolotov, B. Young and R. Ramaraju, "Model and Analysis of Combined Package and on-Chip Power Grid Simulation," in *Proc. Int. Symp. Low-Power Electron. Design*, 2000, pp. 179-184.

[42] H. H. Chen and J. S. Neely, "Interconnect and Circuit Modeling Techniques for Full-chip Power Supply Noise Analysis," *IEEE Trans. Components Packaging Manufacturing Technology, Part B: Advanced Packaging*, vol. 21, no. 3, pp. 209-215, Aug. 1998.

[43] L. R. Zheng, B. X. Li and H. Tenlunen, "Efficient and Accurate Modeling of Power Supply Noise on Distributed On-chip Power Networks," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 2, 2000, pp. 513-516.

[44] J. Choi, M. Swaminathan, D. Nhon and R. Master, "Modeling of Power Supply Noise in Large Chips Using the Circuit-based Finite-difference Time-domain Method," *IEEE Trans. Electromagnetic Compatibility*, vol. 47, no. 3, pp. 424-439, Aug. 2005.

[45] S. R. Nassif and J. N. Kozhaya, "Fast Power Grid Simulation," in *Proc. Design Autom. Conf.*, 2000, pp. 156-161.

[46] Z. Zhu, B. Yao and C. K. Cheng, "Power Network Analysis Using an Adaptive Algebraic Multigrid Approach," in *Proc. Design Autom. Conf.*, 2003, pp.105-108.

[47] H. Qian, S. R. Nassif and S. S. Sapatnekar, "Random Walks in a Supply Network," in *Proc. Design Autom. Conf.*, 2003, pp. 93-98.

[48] S. Bobba and I. N. Hajj, "Simultaneous Switching Noise in CMOS VLSI Circuits," in *Proc. Southwest Symp. Mixed-Signal Design*, 1999, pp.15-20.

[49] M. Nourani, M. Tehranipoor and N. Ahmed, "Pattern Generation and Estimation for Power Supply Noise Analysis," in *Proc. Very Large Scale Integr. (VLSI) Test Symp.*, 2005, pp. 439-444.

[50] D. T. Blaauw, A. Dharchoudhury, R. Panda, S. Sirichotiyakul, C. Oh, T. Edwards, "Emerging Power Management Tools for Processor Design," in *Proc. Int. Symp. Low-Power Electron. Design*, 1998, pp. 274-278.

[51] B. Wang and P. Mazumder, "Bounding Supply Noise Induced Path Delay Variation by a Relaxation Approach," in *Proc. Int. Conf. Very Large Scale Integr. (VLSI) Design*, 2006.

[52] M. Hashimoto, J. Yamaguchi and H. Onodera, "Timing Analysis Considering Spatial Power/Ground Level Variation," in *Proc. ACM/IEEE Int. Conf. Comput.-Aided Design*, 2004, pp. 814-820.

[53] J. J. Liou, A. Krstic, Y. M. Jiang and K. T. Cheng, "Path Selection and Pattern Generation for Dynamic Timing Analysis Considering Power Supply Noise Effects," in *Proc. ACM/IEEE Int. Conf. Comput.-Aided Design*, 2000, pp. 493-496

[54] K. S. Kim, S. Mitra and P. G. Ryan, "Delay Defect Characteristics and Testing Strategies," *IEEE Design Test of Comput.*, vol. 20, no. 5, pp. 8-16, Sept.-Oct. 2003.

[55] Z. Barzilai and B. K. Rosen, "Comparison of AC Self-Testing Procedures," in *Proc. IEEE Int. Test Conf.*, 1983, pp. 89-94.

[56] G. L. Smith, "Model for Delay Faults Based Upon Paths," in *Proc. IEEE Int. Test Conf.*,. 1985, pp. 342-349.

[57] E. S. Park, M. R. Mercer and T. W. Williams, "Statistical Delay Fault Coverage and Defect Level for Delay Faults," in *Proc. IEEE Int. Test Conf.*, 1988, pp. 492-499.

[58] C. W. Tseng and E. J. McCluskey, "Multiple-Output Propagation Transition Fault Test," in *Proc. IEEE Int. Test Conf.*, 2001, pp. 358-366.

[59] J. Savir, "Skewed-Load Transition Test: Part I, Calculus," in *Proc. IEEE Int. Test Conf.*, 1992, pp. 705-713.

[60] S. Patel and J. Savir, "Skewed-Load Transition Test: Part II, Coverage," in *Proc. IEEE Int. Test Conf.*, 1992, pp. 714-722.

[61] J. Savir and S. Patel, "Broad-Side Delay Test," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 13, no. 8, pp. 1057-1064, Aug. 1994.

[62] A. Krstic, Y. M. Jiang, K. T. Cheng, "Delay Testing Considering Power Supply Noise Effects," in *Proc. IEEE Int. Test Conf.*, 1999, pp. 181-190.

[63] A. Krstic, Y.-M. Jiang and K. T. Cheng, "Pattern Generation for Delay Testing and Dynamic Timing Analysis Considering Power-Supply Noise Effects," *IEEE Trans. Comput.-Aided Design*, vol. 20, no. 3, pp. 416-425, Mar. 2003.

[64] A. Kokrady and C. P. Ravikumar, "Fast, Layout-aware Validation of Test-vectors for Nanometer-Related Timing Failures," in *Proc. Int. Conf. Very Large Scale Integr. (VLSI) Design*, 2004. pp. 597-602.

[65] S. T. Zachariah, Y.-S. Chang, S. Kundu and C. Tirumurti, "On Modeling Cross-talk Faults," in *Proc. Design Autom. Test Eur. Conf. Exhibition*, 2003, pp. 490-495.

[66] P. A. Sandborn, M. S. Abadir and C. F. Murphy, "The Tradeoff Between Peripheral and Area Array Bonding of Components in Multichip Modules," *IEEE Trans. Components Packaging Manufacturing Technology*, Part A, vol. 17, no. 2, 249-256, Jun. 1994.

[67] 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," in *Proc. IEEE Very Large Scale Integr. (VLSI) Test Symp.*, 2005, pp. 235-240.

[68] J. Wang, Z. Yue, X. Lu, W. Qiu, W. Shi and D. M. H. Walker, "A Vector-based Approach for Power Supply Noise Analysis in Test Compaction," in *Proc. IEEE Int. Test Conf.*, 2005.

[69] J. Wang, D. M. H. Walker, A. Majhi, B. Kruseman, G. Gronthoud, L. E. Villagra, Paul van de Wiel and S. Eichenberger, "Power Supply Noise in Delay Testing," in *Proc. IEEE Int. Test Conf.*, 2006.

[70] P. van de Wiel, W. Heuvelman, "Theory of decap position in a SoC," *NXP Research*, Technical Note NXP-R-TN 2007/00070.

[71] B. Kruseman, A. K. Majhi, G. Gronthoud and S. Eichenberger, "On Hazard-free patterns for fine-delay fault testing," in *Proc. IEEE Int. Test Conf.*, 2004, pp.213-222.

[72] L. Bisdounis and O. Koufopavlou, "Short-Circuit Energy Dissipation Modeling for Submicrometer CMOS Gates," *IEEE Trans Circuits Syst.*, vol. 47, no. 9, pp. 1350-1361, Sept. 2000.

[73] D. Sylvester and K. Keutzer, "System-level Performance Modeling with BACPAC- Berkeley Advanced Chip Performance Calculator," in *Proc. (workshop notes) ACM Int. Workshop Syst.-Level Interconnect Prediction*, 1999, pp. 109-114.

[74] D. Sylvester, W. Jiang and K. Keutzer, "Berkeley Advanced Chip Performance Calculator," [online]. Available: http://www.eecs.umich.edu/~dennis/bacpac

[75] S. R. Nassif and E. Acar, "Advanced Waveform Models for the Nanometer Regime," in *Proc. ACM/IEEE Workshop Timing Issues*, 2004.

[76] G. Bai, S. Bodda and I. N. Hajj, "Static Timing Analysis Including Power Supply Noise Effect on Propagation Delay in VLSI Circuits," in *Proc. ACM/IEEE Design Autom. Conf.*, 2001, pp. 295-300.

[77] W. G. Cochran and G. M. Cox, *Experimental Designs*, John Wiley & Sons, 1957.

[78] T. M. Niermann, R. K. Roy, J. H. Patel and J. A. Abraham, "Test Compaction for Sequential Circuits," *IEEE Trans. Comput.-Aided Design*, vol. 11, no. 2, pp. 260-267, Feb. 1992.

[79] I. Hamzaoglu and J. H. Patel, "Compact Two-pattern Test Set Generation for Combinational and Full Scan Circuits," in *Proc. IEEE Int. Test Conf.*, 1998, pp. 944-953.

[80] K. O. Boateng, H. Konishi and T. Nakata, "A Method of Static Compaction of Test Stimuli," in *Proc. Asian Test Symp.*, 2001, pp. 137-142.

[81] I. Pomeranz, L. N. Reddy and S. M. Reddy, "COMPACTEST: A Method to Generate Compact Test Sets for Combinational Circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 12, no. 7, pp. 1040-1049, Jul. 1993.

[82] L. N. Reddy, I. Pomeranz and S. M. Reddy, "COMPACTEST-II: A Method to Generate Compact Two-pattern Test Sets for Combinational Logic Circuits," in *Proc. ACM/IEEE Int. Conf. Comput.-Aided Design*, 1992, pp. 568-574.

[83] S. Kajihara, I. Pomeranz, K. Kinoshita and S. M. Reddy, "Cost-effective Generation of Minimal Test Sets for Stuck-at Faults in Combinational Logic Circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 14, no. 12, pp. 1496-1504, Dec. 1995.

[84] B. Ayari and B. Kaminska, "A New Dynamic Test Vector Compaction for Automatic Test Pattern Generation," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 13, no. 3, pp. 353-358, Mar. 1994.

[85] I. Pomeranz and S. M. Reddy, "On Static Compaction of Test Sequences for Synchronous Sequential Circuits," in *Proc. ACM/IEEE Design Autom. Conf.*, 1996, pp. 215-220.

[86] S. T. Chakradhar and A. Raghunathan, "Bottleneck Removal Algorithm for Dynamic Compaction in Sequential Circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 16, no. 10, pp. 1157-1172, Oct. 1997.

[87] I. Pomeranz and S. M. Reddy, "COREL: A Dynamic Compaction Procedure for Synchronous Sequential Circuits with Repetition and Local Static Compaction," in *Proc. Int. Conf. Comput. Design*, 2001, pp. 142-147.

[88] G. Ruifeng, I. Pomeranz and S. M. Reddy, "On Improving Static Test Compaction for Sequential Circuits," in *Proc. Int. Conf. Very Large Scale Integr. (VLSI) Design*, 2001, pp. 111-116.

[89] I. Pomeranz and S. M. Reddy, "Improving the Efficiency of Static Compaction Based on Chronological Order Enumeration of Test Sequences," in *Proc. Asian Test Symp.*, 2002, pp. 61-66.

[90] I. Pomeranz and S. M. Reddy, "Test Enrichment for Path Delay Faults Using Multiple Sets of Target Faults," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 22, no. 1, pp. 82-90, Jan. 2003.

[91] R. Sankaralingam, R. R. Oruganti and N. A. Touba, "Static Compaction Techniques to Control Scan Vector Power Dissipation," in *Proc. IEEE Very Large Scale Integr. (VLSI) Test Symp.*, 2000, pp. 35-40.

[92] M.S. Hsiao, E.M. Rudnick and J.H. Patel, "Fast Algorithm for Static Compaction of Sequential Circuit Test Vectors", in *Proc. Very Large Scale Integr. (VLSI) Test Symp.*, 1997, pp. 188-195.

[93] W. Qiu, D. M. H. Walker, N. Simpson, D. Reddy and A. Moore, "Comparison of Delay Tests on Silicon," in *Proc. IEEE Int. Test Conf.*, 2006.

[94] W. Qiu, J. Wang, D. M. H. Walker, D. Reddy, X. Lu, Z. Li, W. Shi and H. Balachandran, "K Longest Paths Per Gate (KLPG) Test Generation for Scan-Based Sequential Circuits," in *Proc. IEEE Int. Test Conf.*, 2004, pp. 223-231.

[95] W. Qiu, J. Wang and D. M. H. Walker, "At-Speed Test for Path Delay Faults Using Practical Techniques," in *Proc. IEEE Int. Workshop Defect Based Testing*, 2004, pp. 59-64.

[96] J. Wingfield, J. Dworak and M. R. Mercer, "Function-based Dynamic Compaction and Its Impact on Test Set Sizes," in *Proc. IEEE Int. Symp. Defect Fault Tolerance Very Large Scale Integr. (VLSI) Syst.*, 2003, pp. 167-174.

[97] J. Zeng, M. Abadir, G. Vandling, L. Wang, A. Kolhatkar and J. Abraham, “On Correlating Structural Tests with Functional Tests for Speed Binning of High Performance Design,” in *Proc. IEEE Int. Test Conf.*, 2004, pp. 31-37.

VITA

Jing Wang

Computer Science Dept., Texas A&M University,

College Station, TX 77840

E-mail: jingw@tamu.edu

Jing Wang was born in Ningbo, China. She obtained her B.S. in computer science from Zhejiang University, Hangzhou, China in July 2002, and a Ph.D. in computer engineering from Texas A&M University, College Station, Texas, in August 2007. Her research interests are delay testing, automatic test pattern generation, power supply noise and timing analysis. She is a member of IEEE.