

Chapter 2

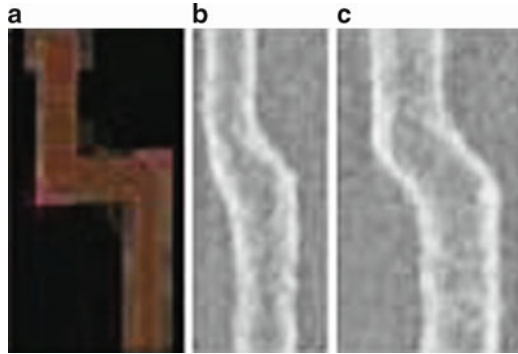
Delay Test and Small-Delay Defects

As technology scales downwards, new challenges are emerging for test engineers. The deep-submicron effects are becoming more prominent with shrinking technology, thereby increasing the probability of timing-related defects [22,24]. As a result, the stuck-at and I_{DDQ} tests alone cannot ensure high quality level of chips, and at-speed test is needed to cover these timing-related defects. In the past, functional patterns were used for at-speed test. However, functional test generation is difficult and time-consuming for large complex designs. As mentioned previously, functional patterns also have pattern count and coverage issues. A cost-effective alternative are the scan-based structural tests generated by at-speed automatic test pattern generators. The transition fault model and path-delay fault model together provide relatively good coverage for delay-induced defects [5, 13, 26, 29].

2.1 Delay Test Challenging

As mentioned above, the transition and path-delay fault models provide better defect coverage, increasing the production test quality and reducing the defective parts per million (DPM) levels. Thus, the transition and path-delay fault models have become very popular in the past decade. The path delay test targets the accumulated delay defects on critical paths in a design and generates test patterns to detect them. Traditionally, static timing analysis (nominal, best-case, or worst-case) is performed to identify the critical paths in the circuit. As technology continues to scale down, more and more delay variations are introduced, which can affect the performance of the target circuit to a large extent. In this situation, the static timing analysis method becomes quite inaccurate since it does not have the capability to fully address these effects. This section briefly describes some of the delay variations in nanometer technology.

Fig. 2.1 An example of process variations' effects on interconnect characteristics. (a) is the interconnect specification in the layout, (b) and (c) are the possible fabricated interconnect on real silicon



2.1.1 Process Variations Effects

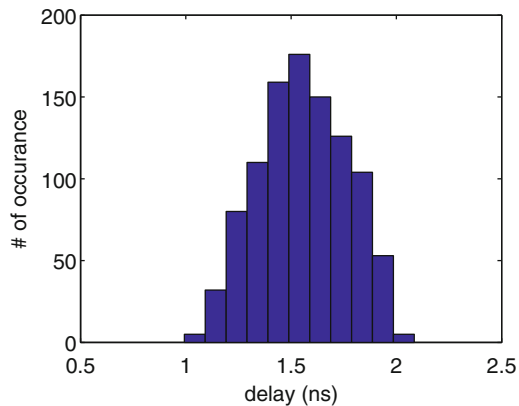
In reality, the parameters of fabricated transistors are not exactly the same as design specifications due to process variations. In fact, the parameters are different from die-to-die, wafer-to-wafer, and lot-to-lot. These variations are systematic and independent in most cases. These variations include impurity concentration densities, oxide thicknesses, and diffusion depths, caused by nonuniform conditions during the deposition and/or the diffusion of the impurities. They directly result in deviations in transistor parameters, such as threshold voltage, oxide thickness, W/L ratios, as well as variation in the widths of interconnect wires [9], and impact the performance (increase or decrease delays) to large extents in the latest technologies.

In applications, designers usually develop process technology files (with nominal, best-case and worst-case conditions) to deal with the variations introduced by manufacturing process to their designs. They then simulate their design in different corner cases specified by these process files to ensure that their design is functional in all corners and that the specific timing behaviors are met for static timing analysis.

Figure 2.1 is an example of the process variations' effects on interconnect characteristics. In this example, the interconnect is specified as (a). However, due to the imperfect fabrication process, the fabricated interconnect could be thinner like (b), or wider like (c). Case (b) produces higher resistance and case (c) may result in a higher coupling capacitance between this interconnect and its neighbor nets. Both cases will affect the interconnect delays, and further variation may have a larger impact on the high-speed designs' performance. Process variations also have a similar impact on transistor characteristics.

In academia, Monte Carlo simulation is often used to emulate the effects of process variation. After 1,000 Monte Carlo simulation runs, it is clear that the delay of this NAND3X1 gate becomes a random variable with a certain distribution, rather than a fixed value even with the same input and output load capacitances (Fig. 2.2). For these 1,000 Monte Carlo simulation runs, only 170 simulations result in the nominal delay of this gate (1.5 ns). The rest of the simulation results are

Fig. 2.2 Monte Carlo simulation results on an NAND3X1 gate (simulation runs: 1,000, load capacitance: 1 pF)



distributed around this nominal delay. In some extreme cases, the delay variation can be 50% compared with the 1.5 ns nominal delay, and the delay variation between the minimum gate delay and maximum gate delay can reach 2X. As technology scales down, the delay variation introduced by process variations can increase even more.

Not only must designers have a full understanding of process variations, taking them into account during design, but so must test engineers. For example, it is important for test engineers to identify and select the timing-critical paths accurately for path-delay pattern generation. As a result of the variability in this process, it is more likely that more paths will become timing sensitive and will require testing. Without considering process variations, they may fail to identify all timing-critical paths for test generation.

2.1.2 Crosstalk Effects

Signal integrity can be affected significantly by the crosstalk effects introduced by parasitic coupling capacitances between parallel interconnects. The crosstalk effects introduced by parasitic coupling capacitances between a target net (the victim) and its neighboring nets (the aggressors) may either increase or decrease the delays on both victim and aggressor nets, depending on the transition direction, transition arrival time, and coupling capacitance between the victim and aggressor nets [27].

Nowadays, more transistors are integrated on a chip. As a result, interconnects have become longer and the interconnect delay has become dominant over the gate delay. As technologies continue to shrink beyond the ultra-deep submicron level, interconnects are also becoming narrower in width. To keep a low wire resistance, the interconnects are becoming taller in height, resulting in large cross coupling capacitances, as shown in Fig. 2.3. It is predictable that in the near future, crosstalk will be a major contributor to the interconnect delay, and will further increase chip delay.

Fig. 2.3 Sidewall capacitances between parallel interconnects for (a) 180 nm technology, and (b) 45 nm technology (for simplicity the resistance is not shown)

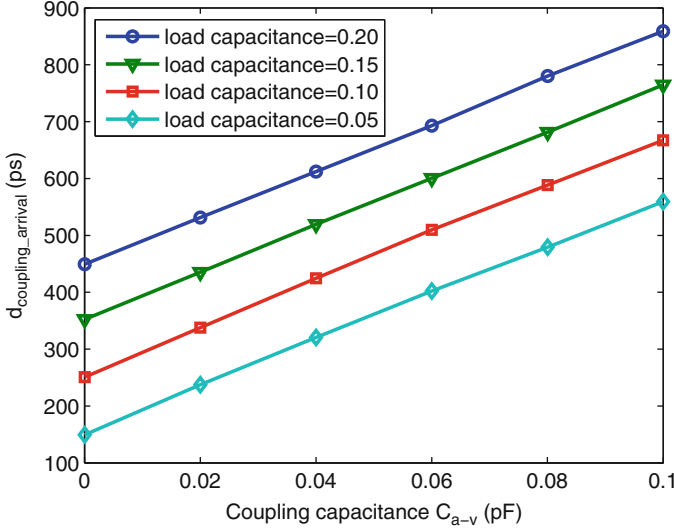
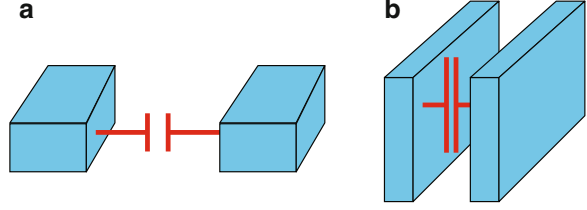
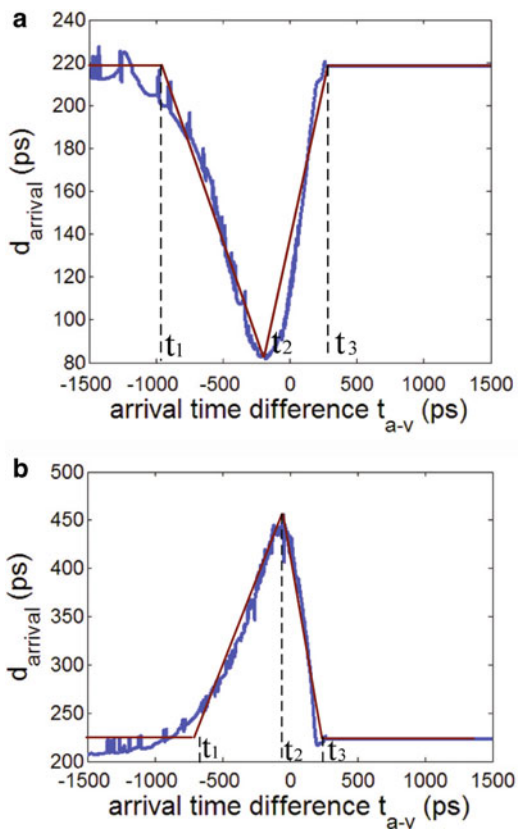


Fig. 2.4 Impact of coupling capacitance on victim propagation delay with same arrival times, opposite transition direction, and different load capacitances. Load capacitance unit is pF

It is necessary for both design and test engineers to analyze and assess crosstalk effects both before signing off on a tape-out and after fabrication (during delay testing). Unfortunately, it is impossible to accurately analyze crosstalk effects without test pattern information. Notice that there may be tens or even hundreds of aggressors for a target victim net. Without test pattern information, there is no way to count how many aggressors are switching with the victim net, and what the active coupling capacitance between the victim net and its aggressors is. The coupling capacitance has a direct impact on the victim's delay, as shown in Fig. 2.4. It can be seen from the figure that for different load capacitance cases, the propagation delay on the victim net increases linearly with the coupling capacitance. $d_{coupling_arrival}$ denotes the victim net delay considering the impact of coupling capacitance size and C_{a-v} is the coupling capacitance between the aggressor and victim nets. For the same transition direction case, the crosstalk delay decreases linearly. In real

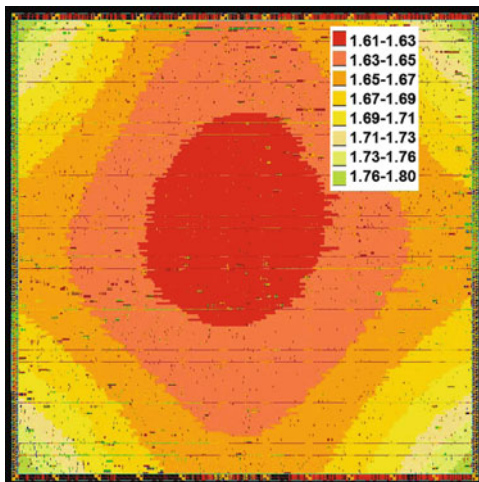
Fig. 2.5 Impact of aggressor arrival time on victim propagation delay when victim and aggressor nets have (a) same transition direction and (b) opposite transition direction. Coupling capacitance: 0.1 pF



applications, the load capacitance of the target victim net (the capacitance between the target net and the substrate) is fixed, but the coupling capacitance of the target victim net depends on its activated aggressors.

Besides coupling capacitance, arrival time and transition direction on the victim and its aggressors can also impact the victim's delay dramatically. Figure 2.5 shows the SPICE simulation results on crosstalk effects between two neighboring interconnects (one victim and one aggressor) with a fixed coupling capacitance. The parameter t_{a-v} denotes the arrival time difference between transitions on the aggressor and victim nets and $d_{arrival}$ represents the victim net delay considering the impact of arrival time difference. It is seen that when the aggressor and victim nets have the same transition direction (see Fig. 2.5a), the victim net will be sped up. Otherwise, the victim net will be slowed down (see Fig. 2.5b). Furthermore, the crosstalk effect on the victim net is maximized when the transition arrival time of aggressor and victim nets are almost the same ($t_{a-v} \approx 0$). Again, without test pattern information, the transitions on the victim and its aggressors cannot be obtained.

Fig. 2.6 IR-drop plot of a test pattern applied to wb_conmax benchmark



2.1.3 Power Supply Noise Effects

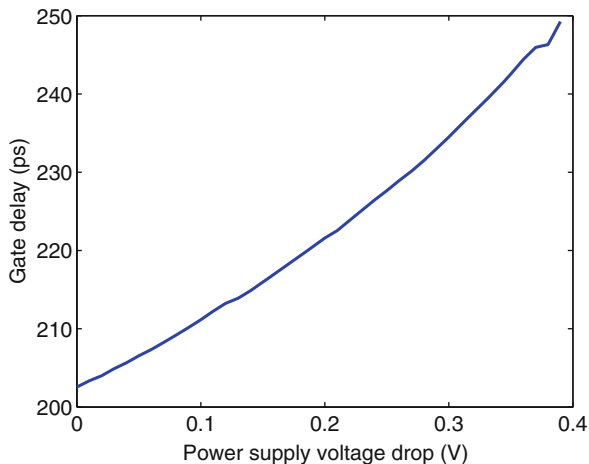
Technology scaling allows us to pack more transistors into one chip and increase the operating frequency of the transistors. This results in increases to both switching and power density. The increase in frequency and decrease in the rise/fall transition times in today's designs causes more simultaneous switching activity within a small time interval and further causes increases in current density and voltage drop along power supply nets. As a result, power supply noise (PSN) has become an important factor for nanometer technology designs.

PSN can be introduced by inductive or resistive parameters, or by a combination of them. Inductive noise is calculated as $L \cdot di/dt$, depending on the inductance L and instantaneous current changing rate. The package leads and wire/substrate parasitics are the main sources of inductive noise. The resistive noise is referred to as IR drop, and depends on the current and distributed resistance in the power distribution network. This book focuses on the resistance-introduced power supply noise, i.e., IR drop, and its impact on circuit performance.

Figure 2.6 shows the IR-drop plot for the wb_conmax benchmark [7] for a randomly selected TDF test pattern. The pattern set for this benchmark is generated using a commercial ATPG tool. The launch-off-capture (LOC) scheme with random-fill is used to minimize the pattern count. This is the average IR drop calculated during the launch-to-capture cycle. Power pads are located in the four corners of the design.

It can be seen that the areas far the from power pads (in the center of the design) have a large IR drop. As shown, different gates in the design will experience different voltage drops, resulting in delay compromise and performance degradation. As the circuit size increases, more severe IR drop is expected in the

Fig. 2.7 Average delay increase of a gate as a result of IR-drop increase (180 nm Cadence generic standard cell library, nominal power supply voltage = 1.8 V)



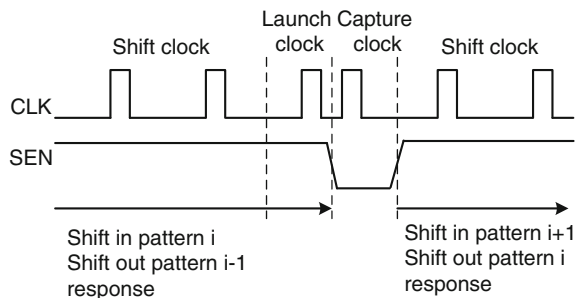
design. These IR-drop plots were obtained from the Cadence SOC Encounter tool and were measured during the at-speed launch and capture cycles of the test pattern. The voltage drop on a gate will directly impact its performance, and further result in performance degradation or functional failures of the circuit. Figure 2.7 presents the simulation results on an AND gate with different power supply voltages. The output load capacitance of the gate is 0.1 pF. It is seen that with 20% IR-drop (0.36 V), the average gate delay decrease can be approximately 21%. This experiment is based on 180 nm Cadence Generic Standard Cell Library with nominal $V_{dd} = 1.8$ V. It has to notice that in smaller technology nodes, the percentage of gate delay increase will be much higher [10]. Note that when more than one gate on a path experiences voltage drop, the performance degradation will be profound.

Power supply noise has been a major issue to deal with when generating at-speed delay test patterns. After scan shifting in the test patterns at a lower frequency, the functional frequency is applied during the launch-to-capture cycle of the at-speed test. In general, the power supply noise during the at-speed delay test is much larger compared with functional circuit operation. This is due to the fact that a larger number of transitions occur within a short time interval in the structural at-speed delay test. Novel frameworks and methods are needed to accurately analyze power supply noise effects for delay fault test pattern generation. This book will present new power supply noise calculations and diagnosis flows for delay test pattern analysis.

2.2 Test for Transition-Delay Faults

As mentioned in the previous chapter, the transition-delay fault (TDF) models a slow signal change defect in the circuit, and that for each fault site, there are two possible faults- *slow-to-rise* and *slow-to-fall*. To test a TDF, a vector pair ($V1$, $V2$)

Fig. 2.8 Waveform for test with the launch-off-shift method



is required. Therefore, there are two test vectors in a single TDF test pattern. A path-delay test pattern also has two test vectors. Vector V1 is called the “initialization vector” and V2 is called the “launch vector.” The response of the CUT to the vector V2 is captured at the operational functional speed. The entire process for testing a TDF can be divided into three cycles:

1. Initialization Cycle, where the CUT is initialized to a particular state by vector V1,
2. Launch Cycle, where a transition is launched at the target gate terminal (V2 is applied);
3. Capture Cycle, where the transition is propagated and captured at an observation point.

Depending on how the transition is launched and captured, there are three transition fault pattern generation methods, referred to as *launch-off-shift* (LOS) or *skewed-load* [12], *launch-off-shift* (LOC) or *broadside* method [11], and *Enhanced Scan* [3]. For the LOS method, the transition at the gate output is launched in the last shift cycle during the shift operation. Then the scan enable (SEN) goes low to enable response capture at the capture clock edge. Figure 2.8 shows the LOS waveform for a scan flip-flop design. It can be seen that LOS requires the SEN signal to be timing critical (must be low between the last shift clock cycle and the capture clock cycle), which may make the DFT design more expensive.

The LOC method does not need at-speed SEN signal. In the LOC method, the launch cycle is separated from the shift operation. At the end of scan-in (shift mode), vector V1 is applied and the CUT is set to an initialized state, and the vector V2 depends on the functional response of the initialization vector V1. As a result, the launch path is less controllable and the test coverage is lower compared with the LOS method. Figure 2.9 shows the LOC waveform for a scan flip-flop design.

The Enhanced Scan technique allows application of any arbitrary vector-pairs by inserting a hold latch between each scan flip-flop. This technique requires that the two vectors V1 and V2 are shifted into the scan flip-flops simultaneously. Using the enhance-scan method, delay tests can be generated by considering the combinational logic alone, which makes the test generation easier. Figure 2.10 shows the architecture of enhance-scan delay test. It can be seen that due to the hold latches, an additional *HOLD* signal is needed.

Fig. 2.9 Waveform for test with the launch-off-capture method

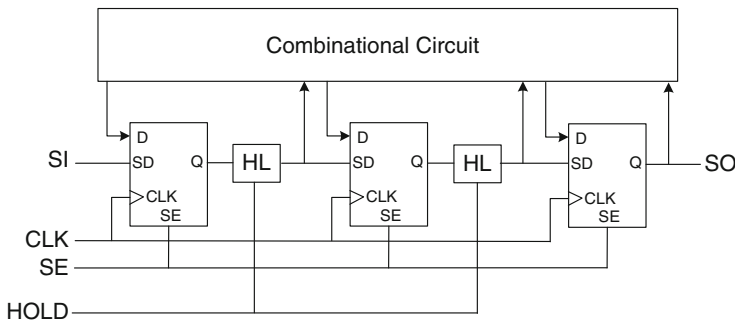
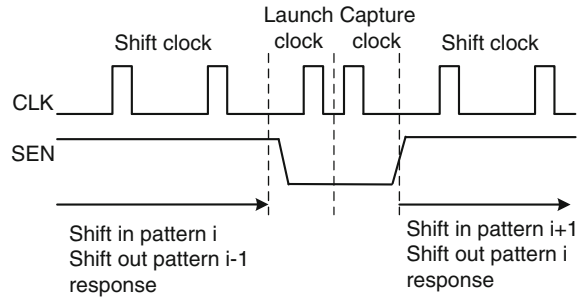


Fig. 2.10 Architecture for the enhanced-scan delay test application

The drawbacks of enhanced scan:

1. It needs to add the area-intensive hold latch and increases the area overhead;
2. The hold latch adds some delay to signal paths and degrades the circuit performance.

2.3 Test for Path-Delay Faults

The path-delay fault assumes that there is a cumulative delay defect along a combinational path, which causes the path to exceed some specified duration. For each combinational path, there are two possible path-delay faults, corresponding to the rising and falling transitions at the input of the path. Two test vectors are required to test each path-delay fault. The first vector $V1$ is used to initialize the target path to a specific state and the second vector $V2$ is used to launch a transition at the input of the target path. The circuit setup needs to ensure that the transition at the input of the path can be propagated to the end of the path.

The terms robust path-delay test and non-robust path-delay test are frequently mentioned in literatures of path-delay test. A robust path-delay test can guarantee

that an incorrect value can be produced at the end of path if the delay of the path under test exceeds a specified duration. A non-robust path-delay test can detect the path-delay fault only when no other path-delay fault is present. Therefore, to effectively detect this path-delay fault, the expected output value must be uniquely controlled by the transition propagating through the target path. Besides the robustness problem, the path number in the circuit is also of large concern, as it is known that the number of paths in the circuit increases exponentially with the circuit size. Therefore, the path-delay fault model is usually only used on a small portion of selected critical paths to generate path-delay test patterns.

2.4 Small-Delay Defects (SDDs)

Small-delay defects (SDD) are one type of timing defect, which introduces a small amount of extra delay to the design. The SDD was firstly alluded to in [4]. Because of their small size relative to the timing margins allowed by the maximum operating frequency of a design, SDDs were not seriously considered in the testing of designs at higher technology nodes. Although the delay introduced by each SDD is small, the overall impact can be significant if the sensitized path is a long/critical path, especially when technology scales to 45 nm and below [21]. As the shrinking of technology geometries and increasing of operating frequency of the design continues, the available timing slack becomes smaller. Therefore, SDDs have a good chance to add enough additional delay to a path to adversely impact the circuit timing and make that part deviate from its functional specifications. Studies have shown that a large portion of failures in delay-defective parts are due to SDDs in the latest technologies [19, 21]. Therefore, SDDs require serious consideration to help increase defect coverage and test quality, or decrease the number of test escapes (i.e., increase in-field reliability), denoted by DPM. Due to the small size of their delay, SDDs are commonly recommended to be detected via long paths running through the fault site.

2.5 Prior Work on SDD Test

2.5.1 *Limitations of Commercial ATPG Tools*

Experiments have demonstrated that TDF test pattern sets can achieve a defect coverage level that stuck-at patterns alone cannot, and they can also detect some SDDs. Unfortunately, such pattern sets have shown a limited ability to detect SDDs in devices and meet the high SDD test coverage requirements in industry, which is very low or close-to-zero DPM for some critical application such as automotive or medical systems. Traditional ATPG tools were developed to target gross delay

defects with a minimal runtime and pattern count, rather than targeting SDDs. To minimize run time and pattern count, TDF ATPGs were developed to be timing unaware and activate TDFs via the easiest sensitization and propagation paths, which are often shorter paths [19]. Note that a transition delay fault can only be detected if it causes a signal transition that exceeds the slack of the affected path. Considering this fact, an SDD may escape the traditional TDF testing and may cause failure in the field if a long path passes through it [1, 19]. Therefore, it is necessary to detect SDDs through long paths.

Commercial timing-aware ATPG tools, e.g., the latest versions of Synopsys TetraMAX [25] and Mentor Graphics FastScan [17], have been developed to deal with the deficiencies of traditional timing-unaware ATPGs. The timing-aware ATPG targets each undetected fault along paths with minimal timing slack, or small timing slack according to user specification, which is the long path running through the fault site. If a fault is detected through a minimal slack path, it is categorized as detected by simulation (DS) and removed from the fault list. The generated pattern will either be filled randomly or fed to the compression engine. Then fault simulation will be run to identify all detected SDDs. If a fault is detected along a path with a slack larger than the specified threshold (least-slack), it is identified as partially detected. Such faults will continue to be targeted every time ATPG generates a new pattern; the fault will be dropped from fault list if, at some point during the pattern generation process, is detected through a path that meets the slack requirement.

The main drawback of timing-aware ATPG algorithms is that they waste a lot of time operating on faults that do not contribute to SDD coverage resulting in a large number of patterns. Experimental results have demonstrated that timing-aware ATPGs will result in significantly larger CPU runtime and pattern count. Furthermore, they seem (1) ineffective in sensitizing large numbers of long paths and (2) incapable of taking into account important design parameters like process variations, crosstalk and power supply noise, which are very important sources for inducing small delay in very deep submicron designs [15, 16].

Due to its underlying algorithms, the n -detect ATPG can also be an effective method for SDD detection, even without timing information of the design. For each target fault, n -detect ATPG will generate patterns trying to detect it n times, through different paths [14, 30]. Therefore, if there is a sensitizable long path running through the target fault site, with a large value of n for the n -detect ATPG, the tool will have a good chance of detecting faults via their possible long paths. In other words, n -detect ATPG can result in high-quality patterns for screening SDDs. Furthermore, experiments have demonstrated that the n -detect ATPG requires much lower CPU runtime when compared with timing-aware ATPG. However, the significantly large pattern count for large n limits the usage of n -detect ATPG in practice.

Figure 2.11 presents the normalized pattern count, number of detected SDDs, and CPU runtime of 1-detect, n -detect ($n = 5, 10, 20$) and timing-aware (ta) pattern sets for the IWLS ethernet benchmark (138,012 gates and 11,617 flip-flops) [7]. The detected SDD is defined as a detected TDF with slack equal or

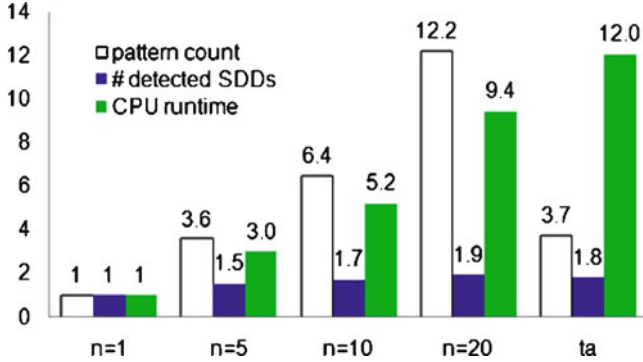


Fig. 2.11 Normalized pattern count, detected SDDs and CPU runtime of different pattern sets for ethernet. The results are normalized with respect to 1-detect pattern set

smaller than $0.3T$, where T is the clock period of the design. It can be seen that n -detect and timing-aware pattern sets can detect more SDDs than the traditional 1-detect timing-unaware pattern set (1.5–1.9X). However, the penalty is a large pattern count (3.6–12.2X) and CPU runtime (3.0–12.0X). It is obvious that as n increases, the increases in pattern count and CPU runtime for the n -detect ATPG are approximately linear. For this design, the timing-aware ATPG results in a pattern count comparable to 5-detect ATPG. However, its CPU runtime is even larger than 20-detect ATPG.

Another method for detecting SDDs is to simply perform faster-than-at-speed testing. This kind of techniques increases the test frequency to reduce the positive slack of the path to detect SDDs on target paths. However, the application of this method is limited since (1) the on-chip clock-generation circuits must be over-designed to meet the requirements of the faster-than-at-speed testing to provide various high frequency steps and frequency sweep ranges, which would make it expensive and difficult to generate given the process variations and (2) the methodology may result in false identification of good chips as faulty due to the reduced cycle time and increased IR-drop [18] leading to unnecessary yield loss.

In summary, current tools such as timing-unaware TDF ATPGs and timing-aware ATPGs are either inefficient at detecting SDDs or suffer from large pattern counts and CPU runtime. Furthermore, none of these methodologies take into account the impact of important design parameters e.g., process variations, power supply noise and crosstalk, which are potential sources of SDDs.

2.5.2 New-Proposed Methodologies for SDD Test

In recent years, several techniques have been proposed for screening SDDs, with most attempts having focused on developing algorithms to target a delay fault via the longest path.

- In [20], the authors proposed an as late as possible transition fault (ALAPTF) model to launch one or more transition faults at the fault site as late as possible to detect the faults through the path with least slack. This method requires a large CPU run time compared to traditional ATPGs.
- A method to generate K longest paths per gate for testing transition faults was proposed in [28]. However, the longest path through a gate may still be a short path for the design, and thus may not be efficient for SDD detection. Furthermore, this method also suffers from high complexity, extended CPU runtime, and pattern count.
- The authors in [2] proposed a delay fault coverage metric to detect the longest sensitized path affecting a TDF fault site. It is based on the robust path delay test and attempts to find the longest sensitizable path passing through the target fault site and generating a *slow-to-rise* or *slow-to-fall* transition. It is impossible to implement this method on large industry circuits since the number of long paths increases exponentially with circuit size.
- The authors in [6] proposed path-based and cone-based metrics for estimating the path delay under test, which can be used for path length analysis. This method is not accurate due to its dependence on gate delay models, unit gate delay, and differential gate delay models, which were determined by the gate type, the number of fan-in and fan-out nets, and the transition type at the outputs. Furthermore, this method is also based on static timing analysis. It does not take into account pattern-induced noise, process variations, and their impact on path delays.
- The authors in [23] proposed two hybrid methods using 1-detect and timing-aware ATPGs to detect SDDs with a reduced pattern count. These methods first identify a subset of transition faults that are critical and should be targeted by the timing-aware ATPG. Then top-off ATPG is run on the undetected faults after timing-aware ATPG to meet the fault coverage requirement. The efficiency of this method is questionable, since it still results in a pattern count much larger than traditional 1-detect ATPG.
- In [19], a static-timing-analysis based method was proposed to generate and select patterns that sensitize long paths. It finds long paths (LPs), intermediate paths, and short paths to each observation point using static timing analysis tools. Then intermediate path and short path observation points are masked in the pattern generation procedure to force the ATPG tool to generate patterns for LPs. Next, a pattern selection procedure is applied to ensure the pattern quality.
- The output-deviation based method was proposed in [16]. This method defines gate-delay defect probabilities (DDPs) to model delay variations in a design. Gaussian distribution gate delay is assumed and a delay defect probability matrix (DDPM) is assigned to each gate. Then, the signal-transition probabilities are propagated to outputs to obtain the output deviations, which are used for pattern evaluation and selection. However, in case of a large number of gates along the paths, with this method, calculated output deviation metric can saturate and similar output deviations (close to 1) can be obtained for both long and intermediate paths (relative to clock cycle). Since in modern designs, there

exists a large number of paths with large depth in terms of gate count, the output deviation-based method may not be very effective. A similar method was developed in [15] to take into account the contributions of interconnect to the total delay of sensitized paths. Unfortunately, it also suffers from the saturation problem.

- A false-path-aware statistical timing analysis framework was proposed in [8]. It selects all logically sensitizable long paths using worst-case statistical timing information and obtains the true timing information of the selected paths. After obtaining the critical long paths, it uses path delay fault test patterns to target them. This method will be limited by the same constraints the path delay fault test experiences.

In general, the most effective way of detecting a SDD is to detect it via long paths. The commercial timing-aware ATPG tools and most previously proposed methods for detecting SDDs have relied on standard delay format (SDF) files generated during physical design flow [17, 25]. However, the path length is greatly impacted by process variations, crosstalk, and power supply noise. A short path may become long because of one or a combination of such effects. Some of these effects may also cause a long path to become short. For instance, in certain circumstance, crosstalk effects can speed up signal propagation and shorten path length. SDF files are pattern-independent and are incapable of taking these effects into consideration. Furthermore, the complexity of today's ICs and shrinking process technologies has made design features more probabilistic. Thus, it is necessary to perform statistical timing analysis before evaluating the path length. Both traditional and timing-aware ATPGs are not capable of addressing these statistical features.

This book presents various hybrid pattern grading and selection methodologies for screening SDDs that are caused by physical defects as well as by delays added to the design by process variations, power supply noise, and crosstalk. From implementations of the presented procedures on both academic and industry circuits, these methods can result in pattern counts as low as a traditional 1-detect pattern set and long path sensitization and SDD detection similar or even better than the n -detect or timing-aware pattern set. This procedure is capable of considering important design parameters such as process variations, crosstalk, power-supply noise, on-chip temperature, Ldi/dt effects, etc. In this book, process variations, crosstalk and power supply noise are added in the pattern evaluation and selection procedures.

2.6 Book Outline

The remainder of the book is organized as follows. Chapter 3 presents a long-path SDF-based hybrid method for screening SDDs and present metrics for pattern evaluation and selection. Chapter 4 adds process variations and crosstalk effects to the pattern evaluation and selection flow. This chapter also evaluated the efficiency

and accuracy of the procedures for process variations and crosstalk calculation. In Chap. 5, the power supply noise and crosstalk-aware hybrid method is presented. This chapter presents a new crosstalk calculation procedure, which is more accurate compared with the methodology used in Chap. 4. A SDD-based hybrid method is presented in Chap. 6, which is very fast and efficient, and can easily be applied to large industry design with millions of gates. Chapters 7 and 8 present the techniques for introducing maximizing crosstalk and power supply noise effects on critical paths, when generating path-delay test patterns. Chapter 9 introduces the fast-than-at-speed test technique. Power supply noise is also considered in this technique. Chapter 10 introduces the techniques for combinational circuit diagnosis, scan chain diagnosis, as well as chip-level diagnosis strategy. Chapter 11 presents a timing-based SDD diagnosis flow.

References

1. A. K. Majhi, V. D. Agrawal, J. Jacob, L. M. Patnaik, "Line Coverage of Path Delay Faults," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, Vol. 8, No. 5, pp. 610–614, 2000
2. A. K. Majhi, V. D. Agrawal, J. Jacob, L. M. Patnaik, "Line coverage of path delay faults," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 5, pp. 610–614, 2000
3. B. Dervisoglu and G. Stong, "Design for Testability: Using Scanpath Techniques for Path-Delay Test and Measurement," in *Proc. Int. Test Conf. (ITC'91)*, pp. 365–374, 1991
4. E. S. Park, M. R. Mercer, T. W. Williams, "Statistical Delay Fault Coverage and Defect Level for Delay Faults," in *Proc. IEEE International Test Conference (ITC'88)*, 1988
5. G. Aldrich and B. Cory, "Improving Test Quality and Reducing Escapes," in *Proc. Fabless Forum, Fabless Semiconductor Assoc.*, pp. 34–35, 2003
6. H. Lee, S. Natarajan, S. Patil, I. Pomeranz, "Selecting High-Quality Delay Tests for Manufacturing Test and Debug," in *Proc. IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems (DFT'06)*, 2006
7. IWLS 2005 Benchmarks, "<http://iwls.org/iwls2005/benchmarks.html>"
8. J. Lion, A. Krstic, L. Wang, and K. Cheng, "False-Path-Aware Statistical Timing Analysis and Efficient Path Selection for Delay Testing and Timing Validation", in *Design Automation Conference (DAC'02)*, pp. 566–569, 2002
9. J. M. Rabaey, A. Chandrakasan, B. Nikolic, "Digital Integrated Circuits, A Design Perspective (Second Edition)," Prentice Hall Publishers, 2003
10. J. Ma, J. Lee, and M. Tehranipoor, "Layout-Aware Pattern Generation for Maximizing Supply Noise Effects on Critical Paths," in *Proc. IEEE VLSI Test Symposium (VTS'09)*, 2009
11. J. Savir and S. Patil, "On Broad-Side Delay Test," in *Proc. VLSI Test Symp. (VTS'94)*, pp. 284–290, 1994
12. J. Savir, "Skewed-Load Transition Test: Part I, Calculus," in *Proc. Int. Test Conf. (ITC'92)*, pp. 705–713, 1992
13. K. Cheng, "Transition Fault Testing for Sequential Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, no. 12, pp. 1971–1983, Dec 1993
14. M. E. Amyeen, S. Venkataraman, A. Ojha, S. Lee, "Evaluation of the Quality of N-Detect Scan ATPG Patterns on a Processor", *IEEE International Test Conference (ITC'04)*, pp. 669–678, 2004
15. M. Yilmaz, K. Chakrabarty, and M. Tehranipoor, "Interconnect-Aware and Layout-Oriented Test-Pattern Selection for Small-Delay Defects," in *Proc. IEEE Int. Test Conference (ITC'08)*, 2008

16. M. Yilmaz, K. Chakrabarty, and M. Tehranipoor, "Test-Pattern Grading and Pattern Selection for Small-Delay Defects," in *Proc. IEEE VLSI Test Symposium (VTS'08)*, 2008
17. Mentor Graphics, "Understanding how to run timing-aware ATPG," Application Note, 2006
18. N. Ahmed, M. Tehranipoor and V. Jayaram, "A Novel Framework for Faster-than-at-Speed Delay Test Considering IR-Drop Effects", in *Proc. Int. Conf. on Computer-Aided Design (ICCAD'06)*, 2006
19. N. Ahmed, M. Tehranipoor and V. Jayaram, "Timing-Based Delay Test for Screening Small Delay Defects," *IEEE Design Automation Conf.*, pp. 320–325, 2006
20. P. Gupta, and M. S. Hsiao, "ALAPTF: A new transition fault model and the ATPG algorithm," in *Proc. Int. Test Conf. (ITC'04)*, pp. 1053–1060, 2004
21. R. Mattiuzzo, D. Appello C. Allsup, "Small Delay Defect Testing," <http://www.tmworld.com/article/CA6660051.html>, Test & Measurement World, 2009
22. R. Wilson, "Delay-Fault Testing Mandatory, Author Claims," *EE Design*, Dec 2002
23. S. Goel, N. Devta-Prasanna and R. Turakhia, "Effective and Efficient Test pattern Generation for Small Delay Defects," *IEEE VLSI Test Symposium (VTS'09)*, 2009
24. S. Natarajan, M.A. Breuer, S.K. Gupta, "Process Variations and Their Impact on Circuit Operation," in *Proc. IEEE Int. Symp. on Defect and Fault Tolerance in VLSI Systems*, pp. 73–81, 1998
25. Synopsys Inc., "SOLD Y-2007, Vol. 1–3," Synopsys Inc., 2007
26. T. M. Mak, A. Krstic, K. Cheng, L. Wang, "New challenges in delay testing of nanometer, multigigahertz designs," *IEEE Design & Test of Computers*, pp. 241–248, May–Jun 2004
27. W. Chen, S. Gupta, and M. Breuer, "Analytic Models for Crosstalk Delay and Pulse Analysis Under non-Ideal Inputs," in *Proc. IEEE International Test Conference (ITC'97)*, pp. 808–818, 1997
28. W. Qiu, J. Wang, D. Walker, D. Reddy, L. Xiang, L. Zhou, W. Shi, and H. Balachandran, "K Longest Paths Per Gate (KLPG) Test Generation for Scan Scan-Based Sequential Circuits," in *Proc. IEEE ITC*, pp. 223–231, 2004
29. X. Lin, R. Press, J. Rajski, P. Reuter, T. Rinderknecht, B. Swanson and N. Tamarapalli, "High-Frequency, At-Speed Scan Testing," *IEEE Design & Test of Computers*, pp. 17–25, Sep–Oct 2003
30. Y. Huang, "On N-Detect Pattern Set Optimization," in *Proc. IEEE the 7th International Symposium on Quality Electronic Design (ISQED'06)*, 2006

Test and Diagnosis for Small-Delay Defects

Tehranipour, M.; Peng, K.; Chakrabarty, K.

2012, XVIII, 212 p., Hardcover

ISBN: 978-1-4419-8296-4