Low-Cost Diagnostic Pattern Generation and Evaluation Procedures for Noise-Related Failures

Junxia Ma[1], Nisar Ahmed[2] and Mohammad Tehranipoor[1]
[1]ECE Department, University of Connecticut, Storrs, CT, 06269
[1]{junxia, tehrani}@engr.uconn.edu
[2]Texas Instruments, Dallas, TX, 75243, n-ahmed@ti.com

ABSTRACT

As technology feature geometries shrink, failures caused by signal integrity issues have become prominent during test. To avoid the time consuming silicon inspection and reduce the engineering cost and effort for failure analysis, a fast and cost-effective diagnostic flow is proposed in this paper. The flow targets delay faults and can be used to (1) identify noise-related failures with a quiet pattern and (2) evaluate the failed pattern in terms of its noise-induced delay to help identify the root cause of failure. A novel procedure is developed to generate a quiet pattern to help differentiate sources of the failure. The quiet pattern targets the same physical defects as the failed pattern but offers much lower noises level. A pattern evaluation procedure is used to evaluate the noise-induced delay. The proposed procedures are implemented on ITC'99 b19 benchmark. Simulation results demonstrate the effectiveness of the proposed procedure in identifying the failure mechanism. The noise-induced path delay for both failed patterns and diagnostic quiet patterns are thoroughly evaluated.

Keywords: *Diagnosis, Delay Test, Crosstalk, Power Supply Noise, Quiet Pattern Generation.*

I. INTRODUCTION

As technology advances, it is vitally important for the semiconductor industry to shorten the time-to-market and deal with yield, yield loss and escape more effectively. Fast and effective fault diagnosis techniques are essential to help improve the yield and product quality [1]. Industry surveys show that more than 70% of all IC designs need one or more respins [2], in spite of the large amount of resources devoted to design, validation and verification at every step. Fast, accurate and low-cost diagnosis is always in need to find the root cause of the failures and provide feedback to the designers.

Timing failures are often the result of a combination of weak points in a design and silicon abnormalities [2], which reduce the noise immunity of the design and expose it to signal integrity (SI) issues. Design weak points are due to imperfect design, i.e., poor power planning, inadequate power vias, and long parallel interconnects; while silicon abnormalities are caused by manufacture errors, i.e., missing vias, via voids, resistive opens or vias for power/ground lines and signal interconnects. A poor power planning, resistive open power lines/vias, or missing power vias can incur on-chip power droop for some test vectors. The power droop can impact a gate(s) on a critical path and may cause timing failure. Crosstalk noises introduced by large parasitic coupling capacitances between long parallel interconnects can also impact the path delay.

Such failures are switching dependent and may only be excited with certain test vectors as inputs. Mostly small-delay defects (SDDs) manifest such problems and the accumulative SDDs cause timing failures. Currently the commercial diagnosis tools are noise unaware. If a test part fails due to excessive power supply noise and/or crosstalk noise, with sufficient fail/pass test patterns counts, the tool can report a list of suspect pins for corresponding failure model (i.e., slow-to-rise/fall faults). However, it has no information of the failure reason for the suspects. Although for pure IR-drop failures, changing test supply voltage can help find the failure reason; it does not help for crosstalk noise related failures, because crosstalk noise has no direct connection with supply voltage. Since no physical defects can be observed under microscope for noise related failures, it is vain to check the suspects under microscope. Besides, laser-based timing analysis [3] can only be performed on device's active diffusion regions; while interconnects usually go through multiple metal layers, so it is very time-consuming and most of the time impossible for failure analysis engineers to inspect the silicon to identify the root cause.

Diagnosis for physical defects such as stuck-at, bridge, short and delay faults have been extensively investigated in the past decade. There are many sophisticated tools and procedures that can effectively point to the location of physical defects based on the collected failure log from tester [4] [5]. There has been less work devoted to developing effective procedures to address noise-related failures. In [6], Killpack et al. discussed the causes of at-speed failures in microprocessors. The relative importance of IR-drop and crosstalk compared with defect issues in observed speed-path failures was addressed. Saxena et al. presented a case study for an IR-drop induced failure in scan-based at-speed test [7]. Mehta et al. proposed a methodology to diagnose delay defects in presence of crosstalk [8]. A test pattern generation method to identify IR-drop failures during launch-off-shift (LOS) test was proposed in [9]. It minimizes launch and capture mode transitions in LOS test to reduce test mode IR-drop; however, it does not take into account crosstalk-induced delay. In practice, Shmoo plots developed based on sweeping the voltage and frequency during test pattern application are used for failure root cause analysis. However, changing frequency and voltage will only change the voltage drop characteristics and other sources of noise is neglected. For instance, as technology scales, crosstalk-induced delay could be as much as IR-drop induced delay in the circuit.

Motivated by the reasons presented above, we propose fast, low-cost diagnostic pattern generation and evaluation procedures that are applicable to both transition delay faults (TDF) and path delay faults (PDF), and can be used in both launch-off-shift (LOS) and launch-off-capture (LOC) test schemes. It avoids the time consuming silicon inspection and reduces the engineering cost and effort in failure analysis. The proposed diagnostic procedures can be used to: (1) Identify the noise caused failures; (2) Evaluate the noise strength for the failed patterns; (3) Grade patterns during path delay fault testing to ensure selection of high quality patterns to improve the quality of manufacturing test and minimize escape.

The rest of the paper is organized as follows. In Section II, an example is used to discuss the problem and application targeted by

* This work is supported by National Science Foundation grant CCF-0811632.

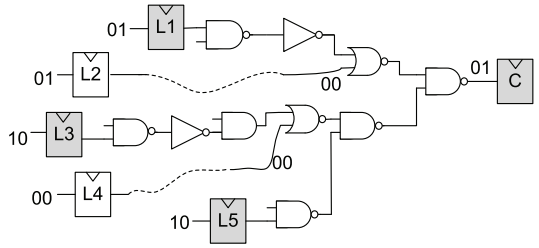


Fig. 1. An example for path delay fault detection.

this work. Section III presents the overall view of the fast, low-cost noise-related failure diagnosis flow. The diagnostic pattern generation is presented in Section IV. Pattern evaluation is covered in Section V. Simulation results to demonstrate the effectiveness and efficiency of the proposed procedures are shown in Section VI. Finally, in Section VII, we present the concluding remarks.

II. Problem Statement

An example of delay fault detection is shown in Figure 1. For the purpose of illustration, assume that when a test pattern is applied, transitions are launched at *L1*, *L2*, *L3* and *L5*. Among them, transitions starting from *L1*, *L3* and *L5* can be propagated to the capture scan flip-flop (SFF) *C* while transition from *L2* is blocked by the combinational logic between *L2* and *C*. Thus, paths *L1* -> *C*, *L3* -> *C*, and *L5* -> *C* are sensitized by the test pattern applied. The following observations are made from the above example:

- Any physical defect and/or excessive noise-induced faults along these paths, which are long enough to cause SFF *C* to capture a wrong value, can be detected by the test pattern applied.
- If a wrong value is captured by SFF *C*, it must be caused by either physical defects, excessive noise on the detectable paths, or both (e.g. a small-delay defect combined with noise).

Assume that SFF *C* is the flip-flop where error is observed. Physical defects on any of the testable paths ending at *C* and associated with the failed test pattern on tester, including both robust and non-robust testable paths, could be the failing reason. The cumulative noise-induced delay along the long path could cause the failure too. There are two scenarios for the *noise-related failure*:

(I) failure is caused by *excessive* noise introduced by bad noisy test patterns.

(II) failure is caused by combined effect of design weak points, manufacture errors and pattern induced noise; and noise is within *acceptable* range, which means it is comparable with (sometimes higher considering design margins) functional mode noise. It is the chip under test that has low noise immunity.

In this paper, the diagnostic procedure we propose can be used to:

- Distinguish noise-related failures from that caused by physical defects.
- Differentiate the two scenarios discussed above if the failure mechanism is identified as noise-related reason. This is done by performing the proposed pattern evaluation procedure and quantify the noises' impact on path delay.

A novel pattern generation procedure is developed to generate a diagnostic pattern (i.e., quiet pattern) to help differentiate the failures caused by noise from those caused by physical defects. The quiet pattern targets the same physical defects as the original failed pattern but introduces much lower noise levels. Once the noise induced failures are identified, a pattern evaluation procedure is used to check each noise level (e.g., power supply noise and crosstalk) of the failed pattern. If the noise level of the failed pattern is within acceptable range, the failure information can be used as feedback to design engineers to help better perform SI analysis, power-distribution network design, and timing closure. If failures are caused by excessive noises, the failed test pattern must be replaced with one or multiple low-noise test patterns.

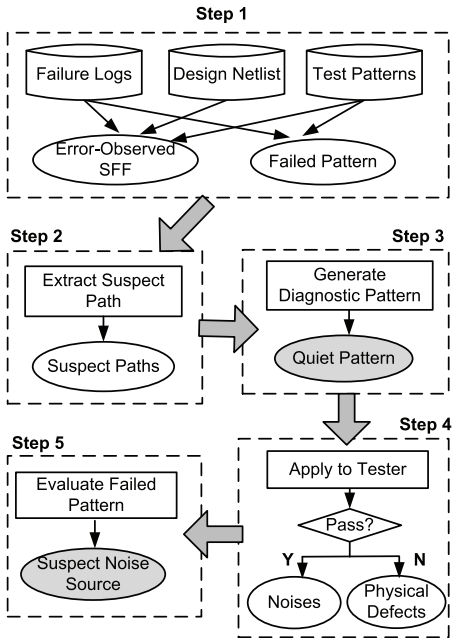


Fig. 2. Proposed flow for noise-related failure diagnosis.

III. Diagnostic Flow

Figure 2 shows the overall noise-related failure diagnosis flow consisting of five major steps, namely (1) Error-observed SFF and failed pattern identification; (2) Suspect paths extraction; (3) Diagnostic pattern generation; (4) Physical/non-physical defects identification; and (5) Failed pattern evaluation. Each step is briefly described in the following.

Step 1: Error-observed SFF and failed pattern identification: The SFF observing error is the capture flip-flop of suspect paths. The input to this step is the failure log collected on tester, the corresponding test pattern set and design netlist. By parsing the failure log, we can easily obtain (i) the failed pattern ID, (ii) observation node, and (ii) observation cycle ID. The pattern ID identifies the failed pattern from the test pattern set; the observation node is the output pin name where the error is observed on the tester. The error-observed SFF can be identified by either custom script or using commercial diagnosis tools.

Step 2: Suspect paths extraction: There are two methods that can be used to extract the suspect paths: (a) By backtracking the simulation pin data from the error-observed SFF, the sensitized paths can be identified as the suspect paths. (b) With the knowledge about error-observed SFF and its corresponding test pattern, we can then extract all the paths ending at the error-observed SFF (also called capture SFF) that are within specified slack threshold. By running fault simulation those paths that are activated during test by the failed pattern can be identified. These paths are taken as initial suspect paths. This is the method we used in this work.

Step 3: Diagnostic (quiet) pattern generation: Diagnostic patterns are generated to differentiate the failure mechanism for the suspect paths. In this paper, the diagnostic pattern is also called "quiet"

pattern because of its low noise property. It can detect the same delay faults on the suspect paths as original failed pattern. That is, the newly generated quiet pattern activates all the suspect physical fault sites in the same way as the failed pattern does. The pattern generation procedure is discussed in details in Section IV.

Step 4: Physical/non-physical defects identification: The quiet pattern, which has noise level much lower than the failed pattern as well as the functional mode, is applied to the tester again under the same test environment (i.e., supply voltage, temperature and frequency). If the quiet test pattern passes on the tester, it proves that the original pattern failed due to noise-related reason, because the only difference between the quiet pattern and the original failed pattern is their noise levels. A pattern evaluation procedure will be performed to quantify the noise induced delay, and further differentiate two noise-related failure scenarios (I) and (II) as discussed in Section II. If the quiet test pattern also fails, then the chip must have failed the original test due to physical defects since the low-noise pattern excludes the noise reason.

Step 5: Failed pattern evaluation: If the quiet pattern passes the test and noise-related reason is identified as the failure mechanism, the failed pattern (here, also called "noisy" pattern) is then examined with our pattern evaluation procedure. The contribution on path-delay increase from noises, such as power supply noise and crosstalk, will be quickly evaluated and represented with pattern quality metrics: Q_{Xtalk} and Q_{PSN}. By comparing the failed pattern's Q values with the mean of the functional patterns, we can conclude whether the failure is caused by bad test patterns with excessive noise (scenario I) or combined effect of design weak points and/or manufacture errors, and pattern induced noise on the chip (scenario II).

IV. Diagnostic Quiet Pattern Generation Procedure

As discussed above, a quiet pattern is needed to distinguish the failure mechanism between physical defects and noises related problems. The quiet pattern should fulfill the following requirements:

(1) It needs to detect all the physical defects in the same way as the failed "noisy" pattern detects on the suspect paths;
(2) It must have a much lower noise level compared with the failed pattern.

If we add path delay faults on the suspect paths and run automatic test pattern generation (ATPG) tool, the tool cannot guarantee to generate a new pattern that detects all the faults. This is because some faults are detected by the don't-care bits, filled randomly, in the original failed pattern. Simulation results show that even if the tool can generate a pattern targeting all the suspect paths sometimes, it cannot fulfill requirement (1) in most cases. The new pattern cannot reproduce the same detection conditions for these physical defects. To solve this problem, *test relaxation* is used to identify the don't-care bits from the failed pattern and re-fill them by a low switching activity filling method.

Test relaxation techniques for different applications have been discussed in [10]-[12]. However, in this work, since the constraint for test relaxation is simpler, which is to detect path delay faults on the suspect paths, don't-care bit identification is easier. We develop a new test relaxation algorithm which runs quite fast on large designs. The proposed algorithm is implemented in a *TCL* script so that it is executable in a commercial ATPG tool.

As can be seen from Figure 3, design netlist and test rule files are read as inputs and path delay faults on suspect paths are taken as the fault candidates. We use the same design rules as that used during original test pattern generation to make sure the test environment for the quiet pattern is the same as the original pattern that failed on the tester. The faults are then analyzed by the ATPG tool. This is the most time consuming part for large designs. However, in this flow, it only needs to be done once. By developing such a *TCL* script and running test relaxation procedure within the ATPG tool, we eliminate the needs to run design compiling, rule checking, and fault analysis for each fault simulation run. This brings a major advantage on the proposed pattern generation method – *it can be performed on large designs very fast*.

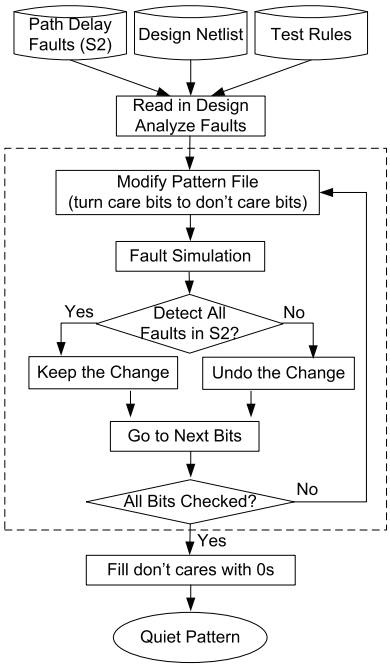


Fig. 3. Diagnostic quiet pattern generation procedure.

During the test relaxation, we tentatively replace care bits with don't-care bits, then based on the fault detection results of fault simulation, we decide whether to keep the changes to the bit value or not. In this way, all the bits for scan and the primary inputs in the pattern file are checked and don't-care bits are identified. A quiet pattern is then generated by filling the don't-care bits with low switching filling methods, fill-0/1/adjacent methods.

The scan chains are processed one by one. As there is always a large number of don't-care bits, for each scan, we replace all the bits with don't-care bits in the first iteration for the selected scan chain. Fault simulation is performed with the modified pattern file. If all the path delay faults are detected, the replacement made in the previous iteration will be accepted; otherwise, the changes will be discarded. In the latter case, the length of bits to be replaced for next iteration will be reduced by half. So a shorter new replacement is made and the process is repeated until all the don't-care bits in the pattern file are identified and replaced. The complexity of this algorithm is $O(Nlogn)$, where *N* is the number of total scan chains, and *n* is the average scan chain length. The entire test relaxation process can finish in a short time for even large designs. The CPU run time results will be presented in Section VI.

V. Pattern Evaluation

Once the failing reason is identified as noise-related by the quiet pattern, it is beneficial to know (1) how much impact each noise source has on path delay and (2) the noise level of each effect. This information is important for identifying the root cause of the failure, i.e., whether it fails due to bad design (weak points and/or manufacture errors) or bad test pattern.

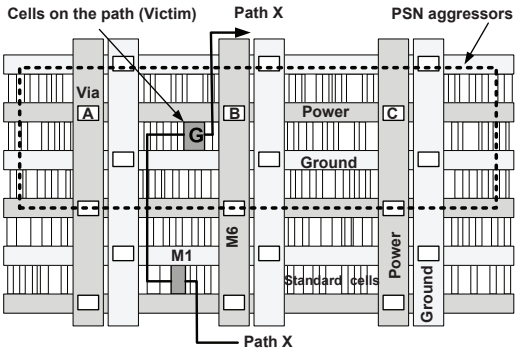


Fig. 4. Aggressor cells identification for power supply noise.

In this work, power supply noise and crosstalk are considered as the two major noise sources that negatively impact signal integrity and can potentially cause silicon failure and escape. A pattern evaluation procedure is used to grade the failed pattern and report the impact of these two noises on path delay. In this phase, the failed patterns refer to those that caused the failure due to noise.

The failed patterns are graded against the suspect paths. Pattern quality metrics Q_{PSN} and Q_{Xtalk} [13] are used to quantify the power supply noise and crosstalk, respectively. The pattern quality metric indicates the amount of noise-induced path delay increase. For each suspect path, we calculate its Q_{PSN} and Q_{Xtalk} values. When comparing these values with that of the functional patterns, we can determine whether the failure is caused by excessive pattern-induced noise or low noise immunity of the chip under test. If the failure is caused by low noise immunity of the design, the pattern evaluation results can help locate the weak point/area in the layout.

A. Aggressor Identification for Power Supply Noise

Simulations have been performed to analyze localized IR-drop effects caused by switching cells that are in close proximity to one another in [14], which shows the closer the neighboring cell is to the switching cell, the larger the voltage drop created by the switching cell and experienced by the neighboring cell. As shown in Figure 4, cell *G* is a cell on path under test. We refer to cell G as "victim" cell and the neighboring cells as "aggressors" because their switching activity can impact the voltage drop and performance of the victim cell. If there are *m* gates on a critical path, we consider all of them to be victim cells; some may be in the same row and others in different rows. As shown in the dashed box in Figure 4, cells directly adjacent to G and those that extend beyond the left and right of *Via A* and *Via C*, respectively, are classified as aggressor cells. The block with dashed lines in Figure 4 shows all the aggressor cells for victim cell G.

B. Aggressor Identification for Crosstalk Noise

To evaluate the impact of crosstalk effects from the aggressors, we need to identify the aggressor nets for the critical paths, and this requires knowledge of the physical design. We use the extracted coupling capacitance of each of the nets to identify those that will have a significant effect on the victim net [15]. A distributed *RC* model is used as the interconnect model during parasitic extraction. We use a minimum coupling threshold during 3D extraction of the layout to prune aggressors with coupling capacitance smaller than the threshold. Using the coupling threshold will reduce the complexity of our analysis by filtering some of the neighboring nets that have almost no effect on the victim path. This will eliminate nets that may be near each other but are routed perpendicularly.

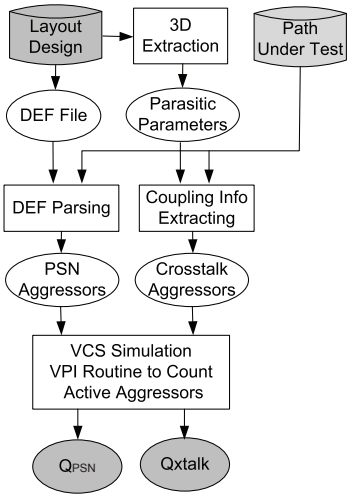


Fig. 5. Pattern evaluation procedure.

C. Pattern Quality Value Calculation

Figure 5 depicts the flow of our pattern evaluation procedure. The inputs are layout design and the suspect paths. For each path under consideration, we can identify its power supply noise aggressors by parsing the Design Exchange File (DEF) file. DEF file is generated from the layout design and contains the physical placement and routing information of all the elements in the circuit. We developed an in-house tool DEF file parser in *C* program. Since the timing analysis tool stores the parasitic coupling information, custom scripts have been developed to report this for each net of the critical path to assist in identifying the crosstalk aggressor nets. A verilog programming language interface (called VPI) routine is used in gate level simulation to monitor the transition of all the aggressors and calculate *Q*'s.

The pattern quality value (Q) for power supply noise is the sum of the weighted switching activity (*WSA*) [16] value of all N_{aggr_cell} aggressor cells, which can be expressed by

$$Q_{PSN} = \sum_{k=0}^{N_{aggr\_cell}} s_k \cdot WSA_k, \text{ where}$$

$$WSA_k = \tau_k + \phi_k f_k \tag{1}$$

$$s_k = \begin{cases} 1, & \text{Transition on gate output} \\ 0, & \text{No transition} \end{cases}$$

The *WSA* of switching gate k depends on its gate weight τ_k, the number of fan-out, f_k, and the fan-out load weight, ϕ_k.

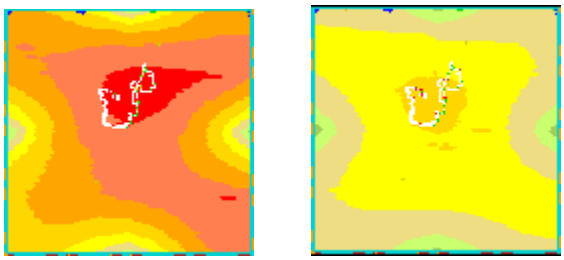
In a similar way, we calculate *Q* for crosstalk noise by

$$Q_{Xtalk} = \sum_{i=0}^{N_{aggr\_net}} d_i \cdot C_i \cdot f(\Delta t), \text{ where}$$

$$d_i = \begin{cases} 1, & \text{Opposite transition} \\ 0, & \text{No transition} \\ -1, & \text{Same transition} \end{cases} \tag{2}$$

$$f(\Delta t) = \begin{cases} 1, & t_1 < \Delta t < t_2 \text{ and } \Delta t = t_a - t_v \\ 0, & \text{otherwise} \end{cases}$$

For all N_{aggr_net} aggressor nets, the equation considers the direction of the transition with respect to the associated nets of the targeted path, d, the amount of coupling between the two nets, C_i, and a timing window $f(\Delta t)$ to take into account the arrival time difference (Δt) of the transitions on aggressors (t_a) and victims (t_v). A positive value of Q_{Xtalk} indicates that the targeted path will experience slow-down due to induced crosstalk effects from the switching aggressors.



(a) (b)

Fig. 6. IR drop plots for (a) failed (noisy) pattern and (b) quiet pattern with the longest suspect path highlighted.

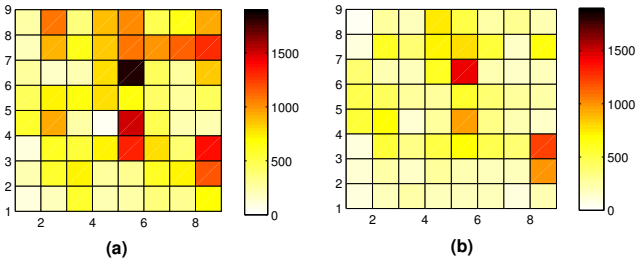


Fig. 7. *WSA* plots for (a) failed (noisy) pattern and (b) quiet pattern. Layout is partitioned into 9x9 regions according to the power distribution network structure.

Similarly, a negative Q_{Xtalk} indicates that the targeted path will experience speed-up.

VI. Results

We implemented the proposed diagnostic pattern generation and pattern evaluation procedures on benchmark *b19*, the largest circuit among ITC'99 benchmarks [17], containing about 200K gates and more than 6K flip-flops. The physical layout was designed using 180nm Cadence Generic Standard Cell Library [18] with 1.8 V as the typical supply voltage.

Because of the lack of test chips and testers, to validate and analyze the effectiveness and complexity of the proposed method, we use simulation to "diagnose" the failures observed at the scan output pins. The effectiveness of the quiet pattern will be demonstrated by comparing its path delay with that of the failed pattern.

For a bit mismatch occurs on a scan output pin on b19, we locate the error-observed SFF using the failed pattern ID, cycle ID and observed pin name, as discussed in Step 1 in Section III. By running a custom developed script with timing analysis tool, we find there are a total of 3843 paths connecting to the capture SFF. We consider both slow-to-rise and slow-to-fall path delay faults for all these paths, and run fault simulation against the failed pattern using ATPG tool. 98 paths out of the 3843 paths are reported as detected; these paths are taken as the suspect paths. We run our test relaxation procedure using the flow presented in Section IV. Out of the 6206 bits in both scan chain inputs and primary inputs, 276 are identified as care bits and the remaining are don't-cares. That is, 95.5% of the input bits can be used for making the pattern a low switching pattern using filling methods. Here, we use 0-fill technique to fill in don't-care bits; other low switching X-filling methods can be easily incorporated in this flow since a large portion of the pattern contains X's. In this way, a quiet pattern is generated such that it ensures detection of the same transition and path delay faults as the failed pattern.

Figure 6 presents the IR-drop plots on power pins for the failed noisy pattern and the diagnostic quiet pattern. In this layout design, four pairs of power and ground pads are placed in the middle of each side. The longest suspect path is highlighted on the plots to show its physical location on the layout. The darkest color in Figure 6(a) represents IR-drop values over 90 *mV* on power pin, which is 5% of the power supply voltage. When comparing the two plots, we can see that the quiet pattern incurs much lower IR-drop in the design and around the suspect path.

TABLE I

Comparison results for noisy and quiet patterns for the longest suspect paths in *b19* benchmark.

Metric	Noisy Pattern	Quiet Pattern	Reduction
Path Delay	7.38 *ns*	6.89 *ns*	6.6% (35.0%[1])
Q_{PSN}	2203	1342	39.1%
Q_{Xtalk}	7.422	1.521	79.5%
IR_{ave}	104.1 *mV*	59.2 *mV*	43.1%
Worst IRdrop	107.8 *mV*	61.0 *mV*	42.3%
Switching Power	67.2*mW*	27.3 *mW*	59.4%
Total Switches	549300	225787	58.9%

[1] This percentage (35%) stands for the reduction on noise-induced path delay.

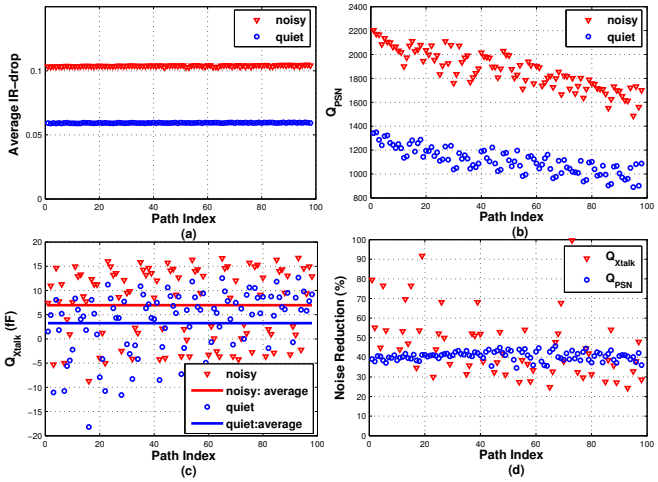


Fig. 8. Comparison between noisy pattern and quiet pattern on 98 suspect paths for (a) average IR-drop; (b) Q_{PSN}; (c)Q_{Xtalk} and (d) Percentage of noise reduction.

IR-drop plots depend both on current distribution and the resistance of the power distribution network. The layout-aware weighted switching activity, i.e., WSA, represents the switching current distribution. Figure 7 illustrates the dynamic WSA plots for both the failed noisy pattern and the diagnostic quiet pattern on b19 layout. We partition the design into 9x9 regions according to the location of power straps and power rings on the layout [16]. We develop a VPI routine and run it with gate-level simulation to calculate the sum of switching gates' WSA for each region. From the two WSA plots it is observed that the quiet pattern considerably reduces the switching current.

Since we do not have a tester to apply the patterns and measure the paths delay, we run fast SPICE simulation to simulate and compare the path delay of the failed and quiet patterns. Table I compares the path delay and other metrics of the longest suspect paths between noisy and quiet patterns. The reduction percentages are listed in Column 4. For path delay, 6.6% reduction is calculated when compared with noisy pattern's path delay; and 35% is the reduction on noise-induced path delay. Noise-induced path delay is calculated by comparing path delays of each case with that of the ideal case 5.98 *ns*, i.e., path delay without noises.

Figure 8 compares average IR-drop IR_{ave}, pattern quality values Q_{PSN} and Q_{Xtalk} of noisy pattern and quiet pattern on 98 suspects paths. The x-axis presents the path index; the paths are sorted based

TABLE II

NOISY AND QUIET PATTERNS COMPARISON FOR LONGEST SUSPECT PATH OF 5 FAILURES IN *b19* BENCHMARK.

Path	Noisy Pattern					Quiet Pattern					X bits
	Delay (*ns*)	IR_{ave} (*mV*)	Q_{PSN}	Q_{Xtalk} (*fF*)	Switch Power (*mW*)	Delay (*ns*)	IR_{ave} (*mV*)	Q_{PSN}	Q_{Xtalk} (*fF*)	Switch Power (*mW*)	(%)
1	8.334	106.9	1194	13.28	135.3	7.815	49.9 (53%)	266 (77%)	3.962 (70%)	83.8 (38%)	98.7
2	8.742	78.6	517	18.88	122.9	8.440	47.0 (40%)	342 (34%)	0.005 (99%)	84.8 (31%)	98.8
3	8.009	105.4	995	5.365	122.1	7.604	50.0 (52%)	265 (73%)	-3.962 (174%)	83.9 (31%)	98.8
4	7.952	102.8	262	-6.230	127.6	7.572	51.0 (50%)	244 (68%)	-13.47 (116%)	81.7 (36%)	98.9
5	5.146	81.0	531	23.20	138.3	4.944	38.7 (52%)	50 (90%)	22.2 (4%)	77.2 (44%)	98.5

on their slack. Path *1* is the longest path with the least slack. From the results shown in Figure 8(a), it can be seen that the quiet pattern reduces the IR_{ave} by more than 40% on all the 98 suspect paths. Since these suspect paths are located close to each other on the layout, in this case, their IR_{ave} values are very close. The pattern evaluation procedure is performed for all the 98 suspect paths. Figure 8(b) shows the power supply noise pattern quality values Q_{PSN} of the suspect paths. Similarly, the crosstalk pattern quality values Q_{Xtalk} are shown in Figure 8(c). The percentage of noise reduction when comparing quiet pattern with noisy pattern for each suspect path is presented in Figure 8(d).

From these figures we can see that, for Q_{PSN}, longer paths reported by static timing analysis tool, that have smaller path index numbers, have higher Q_{PSN} values when compared with short paths shown with large index numbers in Figure 8. Comparing with power supply noise, the Q_{Xtalk} values for the suspect paths are more random. In some cases, short paths may have same amount of crosstalk noise as long paths. This makes sense because crosstalk depends on the routing, the direction of transitions on aggressors as well as the arrival time of the aggressors. The quiet pattern reduces the switching activities around suspect paths, which effectively reduces the aggressor nets and gates' switchings and thus reduces both noises. Figure 8(d) shows significant noise reduction on quiet pattern compared with noisy pattern. The high percentage of noise reduction validates the diagnostic function of the quiet pattern.

We run the proposed pattern generation and evaluation procedures for five other error-observed SFFs. A failure log is generated for five randomly selected SFFs. A quiet pattern is generated for each case. The paths delay of the original failed patterns and the diagnostic quiet patterns are compared. Table II summarizes the comparative performance of the noisy patterns and quiet patterns. The IR_{ave}, Q_{PSN} and Q_{Xtalk} are calculated over the longest suspect paths. The percentages of reduction for each metric are also given in paired parentheses. The large percent of don't-care (X) bits (listed in last column) enables the quiet pattern to reduce the noise effectively.

The simulations were performed on an x86 server architecture, running a Linux OS, 8 CPU cores clocked at 2.826 GHz, and 32GB of RAM. The average CPU run time for quiet pattern generation is 3m22s. The the time to run pattern evaluation procedure over one path is 15.92s; while the time to run it over 98 paths is 16m33s.

VII. CONCLUSIONS

With the increase of power supply noise and crosstalk noise in nanometer designs, diagnosis for noise-related failures during both first silicon and manufacturing test has become an important and challenging task. A fast and low-cost diagnostic flow is proposed in this paper to address this challenging problem. A diagnostic quiet pattern is generated to differentiate the noise-related failures from physical defects. The quiet pattern can detect the same physical defects as the original failed pattern for those that can be captured by the error-observed SFF. A pattern evaluation procedure is also presented in this work to evaluate the failed pattern to quantify the noisy impact on path delay. The information obtained from pattern evaluation can be used to speed up the failure analysis process. The results have demonstrated the effectiveness of the proposed flow in reducing switching noise with the diagnostic quiet pattern and analyzing the patterns that fail due to noise.

REFERENCES

[1] J. A. Waicukauski and E. Lindbloom, "Failure Diagnosis of Structured VLSI," in *IEEE Trans. on Design and Test of Computers*, vol.6, no.4, pp.49-60, August 1989.

[2] R. Aitken and E. J. Marinissen, "Guest Editors' Introduction: Addressing the Challenges of Debug and Diagnosis," in *IEEE Design & Test of Computers*, vol. 25, no. 3, pp. 206-207, May 2008.

[3] J. A. Rowlette and T.M. Eiles, "Critical Timing Analysis in Microprocessors Using Near-IR Laser Assisted Device Alteration (LADA)," in *Proc. of International Test Conference (ITC'03)*, pp. 264-273, 2003.

[4] M. Sharma, W. Cheng, T. Tai, Y. S. Cheng, W. Hsu, C. Liu, S. M. Reddy and A. Mann, "Faster Defect Localization in Nanometer Technology based on Defective Cell Diagnosis," in *Proc. of International Test Conference (ITC'07)*, pp. 1-10, 2007.

[5] V. J. Mehta and K. H. Tsai, "Timing-Aware Multiple-Delay-Fault Diagnosis," in *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28 , no. 2, pp. 245-258, 2009.

[6] K. Killpack, S. Natarajan, A. Krishnamachary, and P. Bastani, "Case Study on Speed Failure Causes in a Microprocessor," in *IEEE Design & Test of Computers*, pp. 224-230, May/June 2008.

[7] 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. of Intl. Test Conf.*, pp. 1098-1104, 2003.

[8] V. J. Mehta, M. Marek-Sadowska, K. H. Tsai and J. Rajski, "Timing Defect Diagnosis in Presence of Crosstalk for Nanometer Technology," in *Proc. of International Test Conference (ITC'06)*, pp. 1-10, 2006.

[9] M. Chen and A. Orailoglu, "Cost-effective IR-drop Failure Identification and Yield Recovery through a Failure-adaptive Test Scheme," in Proc. *Design, Automation, and Test in Europe (DATE'10)*, pp. 63-68, 2010.

[10] S. Kajihara and K. Miyase, "On identifying dont care inputs of test patterns for combinational circuits," in *Proc. Int. Conf. Computer-Aided Design*, pp. 364-369, 2001.

[11] A. El-Maleh and A. Al-Suwaiyan, "An efficient test relaxation technique for combinational & full-scan sequential circuits," in *Proc. 20th IEEE VLSI Test Symp.*, pp. 53-59, 2002.

[12] H. Furukawa , X. Wen, K. Miyase, Y. Yamato, S. Kajihara, P. Girard, L.-T. Wang, and M. Tehranipoor, "CTX: A Clock-Gating-Based Test Relaxation and X-Filling Scheme for Reducing Yield Loss Risk in At-Speed Scan Test," in *Proc. 17th Asian Test Symposium*, pp. 397-402, 2008.

[13] J. Ma, J. Lee, N. Ahmed, P. Girard, and M. Tehranipoor, "Pattern Grading for Testing Critical Paths Considering Power Supply Noise and Crosstalk Using a Layout-Aware Quality Metric," in *Proc. of GLSVLSI'10*, pp. 127-130, 2010.

[14] 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)*, pp. 221-226, 2009.

[15] J. Lee and M. Tehranipoor, "A Novel Test Pattern Generation Framework for Inducing Maximum Crosstalk Effects on Delay-Sensitive Paths," in *Proc. IEEE International Test Conference (ITC)*, pp. 1-10, 2008.

[16] J. Lee, S. Narayan, and M. Tehranipoor, "Layout-Aware, IR-drop Tolerant Transition Fault Pattern Generation," in Proc. *Design, Automation, and Test in Europe (DATE'08)*, pp. 1172-1177, 2008.

[17] http://www.cerc.utexas.edu/itc99-benchmarks/bench.html.

[18] http://crete.cadence.com, 0.18um standard cell GSCLib library version 2.0, Cadence Design System, Inc., 2005.