2010 19th IEEE Asian Test Symposium

A Noise-Aware Hybrid Method for SDD Pattern Grading and Selection

Ke Peng[1], Mahmut Yilmaz[2], Krishnendu Chakrabarty[3], Mohammad Tehranipoor[1]
[1]ECE Department, University of Connecticut
[2]Advanced Micro Devices, Inc., Sunnyvale, California
[3]ECE Department, Duke University

***Abstract*—Testing for small-delay defects (SDDs) is necessary for ensuring product quality in smaller technology nodes. Current tools such as transition-delay fault (TDF) ATPGs and timing-aware ATPGs are either inefficient in detecting SDDs or suffering from large pattern count and CPU runtime. Furthermore, none of these methodologies take into account the impact of pattern-induced noises, e.g., power supply noise (PSN) and crosstalk, which are potential sources of SDDs. In this paper, we present a hybrid method considering the impacts of pattern-induced noises to grade and select the most effective patterns for detecting SDDs. The grading procedure is performed on a large repository of patterns generated by n-detect TDF ATPG. Top-off ATPG is performed after pattern selection to achieve the same fault coverage as that for timing-aware ATPG. The experimental results demonstrate the efficiency of our proposed method; it results in a pattern count close to 1-detect ATPG while sensitizes similar or greater number of long paths than the commercial timing-aware ATPG pattern set.**

***Keywords*-Delay test, small-delay defects, pattern selection, power supply noise, crosstalk**

I. Introduction

Timing-related defects have become major concerns in modern very large scale integration (VLSI) designs [1] [2]. The small-delay defect (SDD) is one type of such timing defects, which can be introduced by imperfect manufacturing process as well as pattern-induced on-chip noises, e.g., power supply noise (PSN) and crosstalk, causing chip failures by introducing extra delay to the design. As technology scales to 45nm and below, resulting in an increase in design density, power density, and operating frequency, SDDs require a serious consideration for ensuring product quality and in-field reliability [3]. Since a SDD only introduces a small amount of extra delay to the design, it is commonly recommended to be detected via long paths running through the fault site.

In practice, most timing defects are targeted by the Transition-delay fault (TDF) model [2]. However, the traditional TDF ATPGs were developed to be timing-unaware, and detect faults via the shorter paths [2] [4]. Therefore, TDF pattern set presents limited capability for detecting SDDs in modern designs. As a result of semiconductor industry demand, commercial timing-aware ATPG tools [5] [6] have been developed to deal with the deficiencies of traditional TDF ATPGs. With timing information, the timing-aware ATPG can detect faults via long paths. However, the significantly increased CPU runtime and pattern count has limited its usage in practice. n-detect TDF ATPG can be considered as an alternate solution for screening SDDs [7] [8]. However, its extremely large pattern count makes it impractical for large industry designs.

With the process technologies scaling down, the impact of pattern-induced noise effects such as PSN and crosstalk must be considered since their impacts on path delays become increasingly significant as operating frequency and power density increase [9]. Due to the complexity of ATPG algorithms, it is very difficult to develop an ATPG tool with all these parasitic effects into consideration. Hence, the pattern selection procedure proposed in this paper can be considered a viable solution.

A. Related Prior Work

Several techniques have been proposed in recent years for screening SDDs and increasing TDF test quality. In [10], the authors proposed an as late as possible transition fault (ALAPTF) model to detect the faults through the path with least slack, which requires extended CPU runtime compared to traditional TDF ATPGs. Two hybrid methods were proposed in [11] using 1-detect and timing-aware ATPGs to detect SDDs based on fault classification. The efficiency of these methods is questioned since it still results in a pattern count much larger than 1-detect TDF ATPG. A static-timing-analysis based method was proposed in [4] to generate patterns that sensitize long paths by masking observation points of short paths and intermediate paths. The output-deviation based method was proposed in [12] and [13]. This method is based on the delay defect probability matrix (DDPM) assigned to each gate, based on which to calculate the output deviations for pattern evaluation and selection. However, this method has an output-deviation saturation problem when there is a large number of gates along the paths.

The impact of power supply noise and crosstalk on circuit performance has been addressed in literature. In [14] and [15], the authors proposed power supply noise aware ATPGs for minimizing the power supply noise on path delays. In [16], an ATPG method was proposed to generate path delay test patterns with maximizing power supply noise effects. All these methods can only be applied to the selected critical paths, since the number of paths in the design increases exponentially with circuit size. The authors in [17] built a look-up table for calculating the propagation delay of the target paths with power supply noise effects. In this work, the authors assumed linear relationship between delay and voltage.

In [19], the authors proposed an analytical model for crosstalk, which was used for generating pattern with crosstalk

[1] The work of K. Peng and M. Tehranipoor was supported in part by SRC under Contract 1587, and by NSF under Grants no. ECCS-0823992 and CCF-0811632. [3]The work of K. Chakrabarty was supported in part by SRC under Contract No. 1588 and by NSF under Grant No. ECCS-0823835.

1081-7735/10 $26.00 © 2010 IEEE
DOI 10.1109/ATS.2010.63

consideration in [20]. This approach considers only single-aggressor case. In [21], a path selection procedure was proposed with coupling noise consideration. The authors in [22] proposed a flow for path-delay pattern generation with maximum crosstalk effects. Similar to the PSN-aware work in [16], this method is only applied to the selected critical paths.

B. Contributions and Paper Organization

In this paper, we propose a noise-aware hybrid method to grade and select the most effective patterns for screening SDDs based on our previous work [18]. The IR2Delay and Xtalk2Delay databases are developed for accurate mapping from IR-drop and crosstalk effects to gate delay compromise, respectively. Our main contributions in the work include:

1. Developing a metric to evaluate each TDF pattern based on its sensitized paths.
2. A procedure is developed to identify all the sensitized paths by each TDF pattern.
3. PSN and crosstalk effects are considered dynamically to accurately evaluate the sensitized paths of each pattern.
4. An efficient pattern selection procedure is developed. The procedure also minimizes the overlap of sensitized paths between patterns (i.e. identifies unique sensitized long paths) to minimize the selected pattern count.

The remainder of this paper is organized as follows. Section II discusses the PSN and crosstalk effects and their impacts on circuit performance. The pattern grading and selection procedure is presented in Section III. The experimental results are presented in Section IV. Finally, we conclude our work in Section V.

II. Analyzing Noise-Induced SDDs

In the section, we will add the impact of PSN and crosstalk effects to our procedure dynamically, when evaluating the delay of sensitized paths by each pattern.

A. Impact of PSN on Circuit Performance

Technology scaling allows packing more transistors into one chip and increasing the operating frequency of transistors. This results in increase in switching and power density. PSN can be introduced by inductive or resistive parameters, or a combination of them. In this work, only resistive noise, which is also referred to as IR drop, and its impact on circuit performance is considered.

Figure 1 shows the IR-drop plot for wb_conmax benchmark [23] for a randomly selected TDF test pattern. The pattern set for this benchmark is generated using a commercial ATPG tool. Launch-off-capture (LOC) scheme with random-fill is used. This is the average IR drop calculated during launch-to-capture cycle. Power pads are located in the four corners of the design. It can be seen that gates in the area far from power pads (center of the design) has a large IR drop. As seen, difference gates in the design will experience different voltage drop, hence delay compromise and performance degradation.

The voltage drop on a gate will directly impact its performance. Figure 2 presents the SPICE simulation results on an AND gate with different power supply voltages. The output load capacitance of the gate is $0.1pF$. It is seen that with 20% IR-drop (0.36V), the average gate delay decrease can be approximately 21%. This experiment is based on 180nm Cadence Generic Standard Cell Library with nominal Vdd=1.8V. We acknowledge that in smaller technology nodes, the percentage of gate delay increase will be much higher [16]. Such effect is represented as an SDD and introduces an extra delay on the path running through this gate. Note that when more than one gate on a path experiences voltage drop, the performance degradation will be profound.

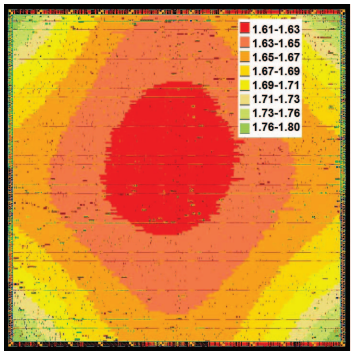

Fig. 1. IR-drop plot of a test pattern applied to wb_conmax benchmark.

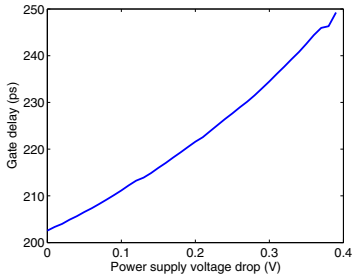

Fig. 2. Average delay increase of a gate as a result of IR-drop increase (180nm Cadence Generic Standard Cell Library, nominal power supply voltage = 1.8V).

We have developed an IR2Delay database based on SPICE simulation to accurately map IR-drop effects to gate delay increase. For each standard cell in the library, we run SPICE simulation and measure its propagation delay with

1) different propagation path, from different input pins to the output pin,
2) different transition direction, including rising and falling transitions,
3) different power supply voltage on the cell, and
4) different load capacitance.

In this paper, we take advantage of the database for PSN-introduced path delay analysis. For each gate along the sensitized path, we (i) obtained its transition pins and transition direction according to the pattern information, (ii) obtained its IR drop using a commercial EDA tool, and (iii) extracted its output load capacitance from layout. With all this information, we search the IR2Delay database for the PSN-introduced delay of the gate, with which to evaluate the sensitized path.

B. Impact of Crosstalk on Circuit Performance

As the technology scales, the distance between interconnects becomes much smaller. This results in a large coupling capacitance and crosstalk effects. Crosstalk effects may either speed up or slow down the delays on both nets, according to the transition direction, transition arrival time, as well as the coupling capacitance [19] [20] [22]. Figure 3 shows the simulation results on crosstalk effects between two neighboring nets with a fixed coupling capacitance. The target net is referred to as victim net and its neighboring net is referred to as aggressor net. It is seen that when the aggressor and victim nets have the same transition direction (see Figure 3(a)), the victim net will be speeded up. Otherwise, the victim net will be slowed down (see Figure 3(b)). Furthermore, the crosstalk effect on the victim net is maximized when the transition arrival time of aggressor and victim nets are almost the same ($t_{a-v} \approx 0$). Figure 4 shows the impact of coupling capacitance on the victim net, with a fixed arrival time difference between transitions on the victim and aggressor nets. The aggressor net has an opposite transition with the victim net in this experiment. It can be seen that the propagation delay on the victim net increases linearly with the coupling capacitance for different load capacitance cases. For the same transition direction case, the crosstalk delay decreases linearly.

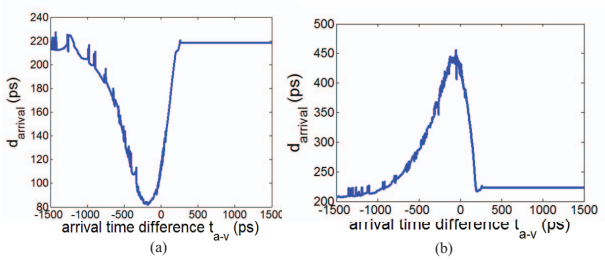

Fig. 3. Impact of aggressor arrival time on victim propagation delay when victim and aggressor nets have (a) same transition direction and (b) opposite transition direction. Coupling capacitance: $0.1pF$.

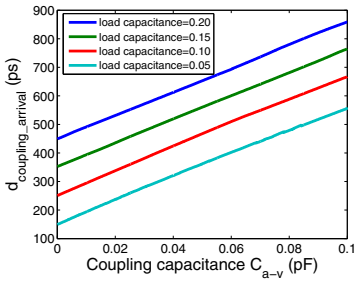

Fig. 4. Impact of coupling capacitance on victim propagation delay with same arrival times, opposite transition direction, and different load capacitances. Load capacitance unit: pF.

Similar to the IR-drop calculation, we developed an Xtalk2Delay database based on SPICE simulation for crosstalk effects calculation. The Xtalk2Delay database is setup based on single-aggressor case. The SPICE simulation is run with

1) different arrival time difference between the victim and aggressor transitions,
2) different load capacitance of the victim and aggressor nets,
3) different coupling capacitance,
4) different transition combinations.

The load and coupling capacitances used for crosstalk calculation are extracted from layout, and the transition direction and arrival time are obtained from the test pattern. For the multiple-aggressor case, we developed a first come first impact (FCFI) procedure to calculate the impact of aggressors one by one. In the FCFI procedure, we first obtain the arrival time of all the sensitized aggressors of the target victim net according to the test pattern, and sort them. Then we apply the impact of the first-coming aggressor using the Xtalk2Delay-based calculation procedure. The second-coming aggressor is applied next. This procedure iterates until the impact of all the sensitized aggressors are applied. This work does not consider the impact of aggressors on each other.

III. Pattern Grading and Selection

A. Sensitized Path Identification and Classification

As mentioned earlier, it is desirable to target SDDs via long paths and gross delay defects via paths of any length. An SDD is defined as a TDF on a long path. Our pattern grading and selection procedure is based on this observation. In the proposed procedure, each TDF pattern is evaluated by its sensitized paths. If a pattern sensitizes a large number of long paths, it is considered as an effective pattern for SDD detection. An in-house tool was developed to report all the sensitized paths of each pattern. For a test pattern, the fault simulation is performed to identify all its detected TDF faults. With this information, we search the topology of the design for the sensitized paths of the pattern. A path is reported as sensitized if all the nodes along the path are detected TDF faults.

A threshold (called long path threshold LP_{thr} in this paper) is needed according to the clock period for path classification. to differentiate the paths to be long or short. In this paper, we define the long path threshold to be $0.7T$, where T is the functional clock period. Thus, a path is considered to be long if its length is equal or greater than this long path threshold. Otherwise, it is considered as a short path. The original path length is obtained from SDF database, and it is dynamically updated based on pattern-dependent PSN and crosstalk calculation. Figure 5 shows the flow of the sensitized path identification and classification. For different test patterns, even the same physical path may have different lengths due to the impact of PSN and crosstalk effects.

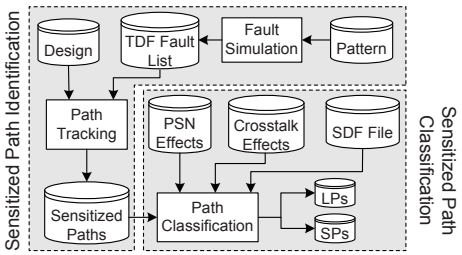

Fig. 5. Sensitized path identification and classification for each pattern.

With the definition of long path threshold, LP_{thr}, we can grade a path by assigning a weight to it according to its path length. Here, we assign a weight of 1.0 for each long path, and a weight of 0.0 for each short path. The test pattern is evaluated by calculating the weight of all its sensitized long paths. Assume that W_{Pi} is the weight of pattern Pi, N is the total number of sensitized paths of pattern Pi, and W_{pathij} is the weight of j-th sensitized path for pattern Pi. The weight of the pattern Pi is calculated by $W_{Pi} = \sum_{j=1}^{N} W_{pathij}$. In this case, the weight of a pattern is equal to the number of its sensitized long paths. We acknowledge that other weight assignments can also be used. It can be varied according to the requirements of the application.

B. Pattern Selection

The path/pattern evaluation procedure in the previous subsection ensures that a pattern sensitizing a large number of long paths will have a large weight. Therefore, we select patterns with the largest weights for screening SDDs. It must be noted that many paths in the design can be sensitized by multiple patterns. In this case, if a path has been sensitized by a previously selected pattern, it is not necessary to use it for evaluating the remaining patterns, if we do not want to target the faults on the path multiple times. Our pattern selection procedure will check the overlap of sensitized paths between patterns, and ensure that only the unique sensitized long paths of a pattern are involved in the pattern evaluation procedure.

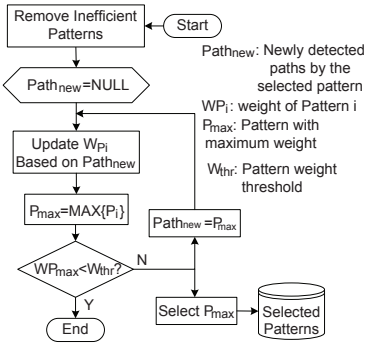

Fig. 6. Sorting and selection procedure for TDF patterns based on their unique weight.

The pattern selection procedure is shown in Figure 6. Before selection, the patterns with no or very few long paths sensitized are removed from the pattern list, since they are inefficient in detecting SDDs, and will never be selected. This can save CPU runtime significantly. Then, the pattern with the largest weight will be firstly selected. All the remaining patterns are re-evaluated by excluding paths detected by the selected pattern. The pattern with largest weight in the remaining pattern set is then selected. This procedure is repeated until some stopping criteria is met, for instance the largest pattern weight evaluated by unique sensitized long paths in the remaining patterns is smaller than a specific threshold. This can further save additional CPU runtime. Experimental results demonstrate that only a small portion of the patterns is selected to sensitize all the long paths that are sensitized by the original pattern repository.

IV. Experimental Results

Our noise-aware hybrid method is shown in Figure 7. In our experiments, n-detect pattern set is used as the original pattern repository. After pattern selection, top-off ATPG is run to meet the fault coverage requirement for the TDF fault model. As a result, the final pattern set of our procedure is the selected pattern set plus the top-off ATPG pattern set.

In this section, we apply our procedure to two IWLS benchmarks [23], namely wb_conmax and ac97_ctrl. The characteristics of these benchmarks are shown in Table I. Note that the data in Table I is obtained after the benchmarks are synthesized by the commercial synthesis tool, and they may be slightly different after placement and routing, since the physical design tool may add some buffers for routing optimization and meeting timing requirements. 180nm Cadence Generic Standard Cell Library was used for physical design.

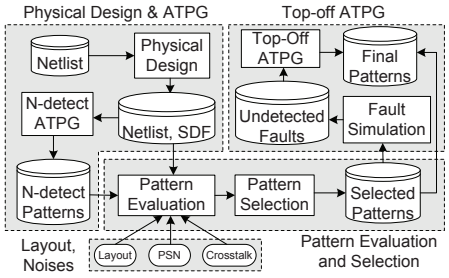

Fig. 7. The entire flow of our noise-aware hybrid method.

TABLE I
Benchmarks Characteristics.

Benchmark	# of gates	# of FFs	Total # of standard cells
wb_conmax	43,219	4,538	47,757
ac97_ctrl	25,488	3,034	28,522

A. Pattern Selection Efficiency Analysis

Table II presents the number and percentage of selected patterns for 90% and 100% long path sensitization, respectively, when applying our pattern selection procedure to different pattern sets of wb_conmax. The long path threshold LP_{thr} in this experiment is $0.7T$.

It can be seen from the table that only 38.01% patterns are needed to sensitize 90% long paths sensitized by the 1-detect pattern set. With the increase in n, the percentage of needed patterns for long path sensitization decreases, even though the number of selected patterns increases. This is because we eliminate the overlap of sensitized paths between patterns in the pattern selection procedure. The table demonstrates that our pattern selection procedure is effective in selecting high-quality SDD patterns. For example, only 1,929 patterns (14.22%) and 3,847 patterns (28.37%) from 13,562 10-detect patterns are selected to sensitize 90% and 100% of all the sensitized long paths, respectively. Note that, in this work, we select patterns from 10-detect pattern set, with 100% long path sensitization, for screening SDDs.

TABLE III
COMPARISON BETWEEN PATTERN COUNT, NUMBER OF SENSITIZED LONG PATHS, AND NUMBER OF DETECTED SDDS. LP_THR = 0.7T.

Benchmark		n=1	n=3	n=5	n=8	n=10	ta	sel.	topoff	sel.+topoff
wb_ conmax	# Patt.	1,931	4,612	7,151	11,028	13,562	19,766	3,847	92	3,939
	# LPs	12,867	19,836	22,933	26,417	27,896	28,012	27,896	43	27,939
	# SDDs	25,368	29,988	31,997	33,664	34,595	34,653	34,595	51	34,646
ac97_ ctrl	# Patt.	1,032	2,335	3,570	5,260	6,393	4,087	184	755	939
	# LPs	532	548	601	658	652	580	652	1	653
	# SDDs	5,894	5,994	6,264	6,711	6,747	6,297	6,747	14	6,761

TABLE II
NUMBER AND PERCENTAGE OF SELECTED PATTERNS FOR LONG PATH SENSITIZATION APPLIED TO WB_CONMAX.

n-detect	Ori. Patt. Set	90% LP Sensitization		100% LP Sensitization	
		#sel. Patt.	% sel. Patt.	#sel. Patt.	%sel. Patt.
n=1	1,931	734	38.01%	1,410	73.02%
n=3	4,612	1,244	26.97%	2,451	53.14%
n=5	7,151	1,655	23.14%	3,052	42.68%
n=8	11,028	1,889	17.13%	3,589	32.54%
n=10	13,562	1,929	14.22%	3,847	28.37%

B. Pattern Set Comparison

Table III compares different kind of pattern sets in terms of pattern count, sensitized unique long paths, and detected unique SDDs for IWLS benchmarks wb_conmax and ac97_ctrl. The pattern selection is performed based on 10-detect pattern set. The pattern weight threshold is set to make sure that the select patterns that can sensitize all the long paths sensitized by the original pattern set. It is clear that as n increases, the number of original patterns, as well as their sensitized long paths and detected SDDs increase for both benchmarks. Timing-aware ATPG pattern set can sensitize more long paths and detect more SDDs compared with timing-unaware 1-detect ATPG pattern set. However, it also results in a pattern count significantly larger than 1-detect, and even larger than 10-detect pattern set for wb_conmax.

Since our selected pattern set is a subset of the original pattern repository, the number of its sensitized long paths is upper bounded by the original pattern repository. It cannot sensitize more long paths than the original pattern repository. However, due to our pattern selection threshold definition, it sensitizes all the long paths of the original pattern repository. Therefore, with top-off patterns, which may incidentally sensitize some extra long paths, our final pattern set can sensitize more long paths and detect more SDDs than the original pattern repository (10-detect pattern set in our experiments) and timing-aware pattern set as in ac97_ctrl. For wb_conmax, the sensitized long paths and detected SDDs of our final pattern set is very close to timing-aware pattern set. When comparing the pattern counts, it is interesting that our final pattern count is much smaller (by 5X) than the original 10-detect pattern repository or timing-aware pattern set, and sometime even smaller than 1-detect pattern pattern set as in ac97_ctrl.

C. The Impact of PSN and Crosstalk

To present the impact of PSN and crosstalk on path delay, we run our procedure (i) without considering noise impacts, (ii) only considering crosstalk effects, (iii) only considering PSN effects, and (iv) considering both crosstalk and PSN effects on a selected number of paths. Table IV shows the path delay for four sample paths for the above four cases, when applying 1-detect pattern set to the wb_conmax benchmark. It is seen that the crosstalk effects (Column 3, shown as "xtalk") can either speed up the path delay as in $Path1$ and $Path2$ or slow down the path delay as in $Path3$, and $Path4$. However, the PSN effect (Column 4, shown as "PSN") always slows down the sensitized path delay. The combination of these two effects (Column 5, shown as "xtalk+PSN") will slow down the target path, even though crosstalk may speed up some paths, since PSN effect seems to be more dominant for the 180nm technology nodes. However, for smaller technology nodes, we expect that crosstalk and PSN present comparable impact on path delay.

TABLE IV
THE IMPACT OF PSN AND CROSSTALK ON PATH DELAY FOR FOUR RANDOMLY SELECTED PATHS IN WB_CONMAX.

Delay (ps)	ori.	xtalk	PSN	xtalk+PSN
Path1	2030.6ps	2026.7ps	2039.7ps	2035.9ps
Path2	2705.8ps	2768.8ps	2861.9ps	2928.6ps
Path3	2075.5ps	2095.2ps	2200.7ps	2221.6ps
Path4	1893.4ps	1884.1ps	1972.5ps	1962.9ps

TABLE V
THE IMPACT OF PSN AND CROSSTALK EFFECTS ON THE PATTERN SELECTION RESULTS FOR WB_CONMAX.

wb_conmax	ori.	xtalk	PSN	xtalk+PSN
# sel. Patt.	1,240	1,335	1,350	1,410
# LPs	9,130	10,060	11,947	12,867
# SDDs	19,724	21,493	23,679	25,368

Table V shows the pattern selection results in the above four cases, when applying 1-detect pattern set to the benchmark wb_conmax. It is clearly seen that the number of long paths in the design increases as a result of the noise impact. Therefore, more patterns are selected for the long path sensitization and SDD detection. In fact, there are $2,099$ short paths that become long as a result of crosstalk slow down effects. At the same time, $1,169$ long paths become short due to crosstalk speed up for this pattern set. Therefore, $2,099 - 1,169 = 930$ new long paths are brought in with crosstalk effects as can be seen in Row 3 for Columns 2 and 3. The PSN can only slow down the path, and it makes in $11,947 - 9,130 = 2,817$ short paths to be long, i.e., become longer than LP_{thr}. Note that the sum of unique long paths caused by crosstalk and PSN effects ($930 + 2817 = 3,747$) is not exactly equal to the results considering both effects ($12,867 - 9,130 = 3,737$). This is because there are few paths that have been slowed down by both effects large enough to make them longer than LP_{thr}, i.e., the procedure only reports the new unique long paths.

D. CPU Runtime Analysis

Table VI presents the CPU runtime for pattern generation with commercial ATPG tool (Row 3, shown as "CPU (ATPG)") and for implementing our hybrid method (pattern grading and selection after sensitized paths identification) on different pattern sets for wb_conmax benchmark. The hybrid

TABLE VI
CPU RUNTIME OF DIFFERENT PATTERN SETS FOR WB_CONMAX BENCHMARK.

pat. sets	1-detect	3-detect	5-detect	8-detect	10-detect	ta
# patt.	1,931	4,612	7,151	11,028	13,562	19,766
CPU(ATPG)	42s	1m43s	2m43s	4m19s	5m24s	34m15s
CPU(HB)	1m43s	2m40s	3m18s	4m8s	4m32s	3m55s
CPU(HB+xtalk)	45m50s	1h50m24s	2h47m20s	4h22m18s	5h21m28s	5h40m33s
CPU(HB+PSN)	9m16s	13m59s	17m48s	19m34s	24m8s	26m40s
CPU(HB+xtalk+PSN)	54m36s	2h0m25s	3h0m24s	4h37m3s	5h38m52s	5h59m21s

method is run (i) without considering PSN and crosstalk noises (Row 4, shown as "CPU (HB)"), (ii) considering only crosstalk effect (Row 5, shown as "CPU (HB+xtalk)"), (iii) considering only PSN effect (Row 6, shown as "CPU (HB+PSN)"), and (iv) considering both crosstalk and PSN effects (Row 7, shown as "CPU (HB+xtalk+PSN)"). From the table, it can be seen that the CPU runtime of n-detect timing-unaware ATPG is small compared to timing-aware ATPG (TA-ATPG) even though n increases up to 10. The CPU runtime of our hybrid method increases approximately linearly with the pattern count of the original pattern repository. Without noise calculation (Row 4, CPU (HB)), our hybrid method is much faster than the timing-aware ATPG. The noise calculation, especially for crosstalk, consumes most of CPU runtime. For example, applying the hybrid method alone to 10-detect pattern set of this circuit consumes less than 5 minutes but it requires more than 5 hours when considering crosstalk (Column 6). This is due to the fact that (i) each sensitized path has multiple net segments, (ii) each net segment may have multiple aggressors, and (iii) the procedure must extract all the information of the victim and aggressors (arrival time, transition direction, load capacitance, as well as coupling capacitance between the victim and aggressors) for accurate crosstalk calculation. The PSN calculation only takes individual gate into account, which makes the procedure much simpler. Therefore, PSN calculation consumes much less CPU runtime compared with crosstalk calculation (24 minutes vs. 5 hours and 21 minutes for 10-detect pattern set as shown in Column 6).

The CPU runtime can be used as trade-off with calculation accuracy. For example, we can ask the procedure to calculate the PSN and crosstalk effects only on critical long paths, which can save significant runtime since only a smaller number of paths in the design are critical. Furthermore, better programming, optimized data structures and algorithms can help speed up the procedure significantly.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a hybrid method for SDD pattern grading and selection considering the impact of pattern-induced noises, e.g., power supply noise and crosstalk. Our pattern selection procedure is performed based on n-detect pattern sets, taking advantage of low CPU runtime and effective SDD detection of the n-detect ATPG. The procedure reduces the pattern count significantly by selecting the most effective patterns in the pattern repository for SDD detection. The method was implemented on two IWLS benchmarks and the experimental results demonstrate the efficiency of our procedure. As for the future work, we will (i) continue to optimize the procedure, and (ii) implement the flow on larger IWLS and ITC'99 benchmarks such as ethernet and b19.

REFERENCES

[1] J. Savir and S. Patil, "On Broad-Side Delay Test," in *Proc. VLSI 2 Symp. (VTS'94)*, pp. 284-290, 1994.

[2] M. Bushnell and V. Agrawal, "*Essentials of Electronics Testing*", Kluwer Publishers, 2000.

[3] R. Mattiuzzo, D. Appello C. Allsup, "Small Delay Defect Testing," *http://www.tmworld.com/article/CA6660051.html* Test & Measurement World, 2009.

[4] N. Ahmed, M. Tehranipoor and V. Jayaram, "Timing-Based Delay Test for Screening Small Delay Defects," *IEEE Design Automation Conf.*, pp. 320-325, 2006.

[5] Synopsys Inc., "SOLD Y-2007, Vol. 1-3," Synopsys Inc.,ct., 2007.

[6] Mentor Graphics, "Understanding how to run timing-aware ATPG," Application Note,2006.

[7] 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.

[8] Y. Huang, "On N-Detect Pattern Set Optimization," in *Proc. IEEE the 7th International Symposium on Quality Electronic Design (ISQED'06)*, 2006.

[9] A. H. Ajami, K. Banerjee, A. Mehrotra, and M. Pedram, "Analysis of IR-Drop Scaling with Implications for Deep Submicron P/G Network Designs," in *Proc. Of the Fourth International Symposium on Quality Electronic Design (ISQED'03)*,pp. 35-40, 2003.

[10] P. Gupta, and M. S. Hsiao, "ALAPTF: A new transition fault model and the ATPG algorithm," in *Proc. Int. Test Conf. (ITC'04)*, 2004.

[11] 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.

[12] 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.

[13] M. Yilmaz, K. Chakrabarty, and M. Tehranipoor, "Interconnect-Aware and Layout-Oriented Test-Pattern Selection for Small-Delay Defects," in *Proc. Int. Test Conference (ITC'08)*, 2008.

[14] N. Ahmed, M. Tehranipoor, and V. Jayaram, "Supply Voltage Noise Aware ATPG for Transition Delay Faults," in *25th IEEE VLSI Test Symmposium (VTS'07)*, 2007.

[15] N. Ahmed, M. Tehranipoor, and V. Jayaram, "Transition Delay Fault Test Pattern Generation Considering Supply Voltage Noise in a SoC Design," in *Proc. Design Automation Conference (DAC'07)*, 2007.

[16] 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.

[17] J. Wang, D. M. Walker, A. Majhi, B. Kruseman, G. Gronthoud, L. E. Villagra, P. van de Wiel, and S. Eichenberger, "Power Supply noise in Delay Testing," in *IEEE International Test Conference (ITC'06)*, 2006.

[18] K. Peng, J. Thibodeau, M. Yilmaz, K. Chakrabarty, and M. Tehranipoor, "A Novel Hybrid Method for SDD Pattern Grading and Selection," in *IEEE VLSI Test Symposium (VTS'10)*,pp. 45-50, 2010.

[19] W. Chen, S. Gupta, and M. Breuer, "Analytic Models for Crosstalk Delay and Pulse Analysis Under Non-Ideal Inputs," in *Proc. of International Test Conf. (ITC'97)*, pp.809-818, 1997.

[20] W. Chen, S. Gupta, and M. Breuer, "Test Generation for Crosstalk-Induced Delay in Integrated Circuits," in *Proc. of International Test Conf. (ITC'99)*, pp. 191-200, 1999.

[21] R. Tayade, J. A. Abraham, "Critical Path Selection For Delay Test Considering Coupling Noise," in *IEEE 13th European Test Symposium (ETS'08)*, 2008.

[22] J. Lee and M. Tehranipoor, "A Novel Test Pattern Generation Framework for Inducing Maximum Crosstalk Effects on Delay-Sensitive Paths," in *IEEE International Test Conference (ITC'08)*, 2008.

[23] IWLS 2005 Benchmarks, "http://iwls.org/iwls2005/benchmarks.html".

[24] ISCAS 89 Benchmarks, "http://www.fm.vslib.cz/ kes/asic/iscas/".