Pattern Generation for Understanding Timing Sensitivity to Power Supply Noise

Tengteng Zhang · Yukun Gao · D. M. H. Walker

Received: 31 July 2014 / Accepted: 7 December 2014 / Published online: 20 December 2014

Abstract Timing prediction has become more and more difficult with shrinking technology nodes. Combining the pre-silicon delay model with post-silicon timing measurements has the potential to improve the accuracy of timing analysis. In this work, we address the problem of automatic test pattern generation for understanding circuit timing sensitivity to power supply noise (PSN) during post-silicon validation. Long paths are selected from a pseudo functional test set to span the power delivery network. To determine the sensitivity of timing to on-chip noise, the patterns are intelligently filled to achieve the desired PSN level. Our previous PSN control scheme is enhanced to consider both spatial and temporal information for better correlation with functional PSN. These patterns can be used to understand timing sensitivity in post-silicon validation by repeatedly applying the path delay test while sweeping the PSN experienced by the path from low to high.

Keywords Delay testing · Pseudo functional test · Power supply noise · Post-silicon validation

1 Introduction

Power supply noise significantly impacts the timing performance of integrated circuits, and it may cause maximum operating frequency (FMAX) mismatch between structural at-speed test and functional test [11]. Much research has been done to generate PSN-aware test patterns. These approaches can be divided into three categories.

The first category spares no effort to minimize power or PSN to avoid over-testing. Many techniques, such as low-power test generation [13], X-filling [10], and power-aware compaction [12], have been developed since low-power design has become mainstream. The disadvantage of these methods is that they do not change the fact that illegal states may still reside in the test pattern. Moreover, patterns that are too quiet will fail to exercise the circuit to the functional mode noise level, which may result in a measured FMAX that is too high, leading to a field failure. Pseudo functional test (PFT) [7] is another technique falling into this category, explicitly placing logic constraints on automatic test pattern generation (ATPG) tools to avoid illegal states.

The second category attempts to maximize PSN in order to achieve high small delay fault coverage and avoid under-testing. In [5], an iterative procedure based on a genetic algorithm (GA) is proposed. In each iteration, waveform simulations and fitness calculation are used to guide selection, crossover and mutation to find patterns that induce larger PSN. In [4], PSN maximization is modeled as a MIN-ONE problem and a SAT solver is used to maximize the transition count. A SAT-based method can find a near-optimal solution. The approach in [8] utilizes a commercial ATPG engine to sensitize as many transitions on neighboring lines as possible. This approach features good compatibility with existing commercial tools, but the test optimality depends on the ATPG algorithm implemented by the tool.

The third category argues that chips should be tested under the maximal functional PSN and attempts to strike a balance between over-testing and under-testing by considering both functional constraints and PSN simultaneously. These

Responsible Editor: T. Xia

T. Zhang (✉) · Y. Gao · D. M. H. Walker
Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77843, USA
e-mail: tengzhang@tamu.edu

Y. Gao
e-mail: Gaoyukun@cse.tamu.edu

D. M. H. Walker
e-mail: walker@cse.tamu.edu

approaches usually consist of two steps – pseudo functional test generation and power supply noise maximization, such as MAX-Fill [3] and PSN-aware PFT [15].

The above approaches can potentially improve delay test quality and achieve high FMAX accuracy. However, they lack the ability to provide knowledge to the design engineer for power supply noise prediction, such as the sensitivity of timing to PSN. Commercial tools can support dynamic IR-drop analysis and the voltage at each node can be back-annotated for accurate circuit simulation [2]. However, this is not an ideal solution considering the shrinking product development window.

Combining the pre-silicon delay model with post-silicon timing measurements has the potential to improve the accuracy of timing analysis since the impact of real silicon variations, such as process variation and power supply noise, is naturally considered. Two typical measurement techniques are critical path monitors (CPM) [9] and ring oscillators (ROs) [6]. They are able to capture the post-silicon variations, including process variation, temperature and voltage. These techniques can have high area overhead, especially for large circuits where large numbers of CPMs and ROs are needed. Another technique to capture the post-silicon process variation is gate-level timing extraction [19]. A novel path selection algorithm is used to generate paths and obtain an accurate variation distribution with no hardware overhead. However, power supply noise is not considered.

In this work, we address the problem of automatic test pattern generation for extracting circuit timing sensitivity to power supply noise during post-silicon validation. Test patterns targeting the K longest paths through each gate are first generated. Then a layout-aware path selection algorithm is implemented to select long paths, which fully span the power delivery network. Finally, the selected patterns are intelligently filled to bring the PSN to a desired level. These patterns can be used to understand timing sensitivity in post-silicon validation by repeatedly applying the path delay test while sweeping the PSN experienced by the path from low to high. This work extends our previous work [16] by including layout-aware path selection and detailed PSN control analysis at the on-path gate level. Previously we only presented experimental results at the path level.

The rest of the paper is organized as follows. In section 2, the pattern generation flow is described in detail. The IR-drop estimation with timing information incorporated is also discussed in this section. In section 3, the experimental results are presented and analyzed. Section 4 presents conclusions and discusses future work.

2 Pattern Generation for Timing Validation

The flow of pattern generation for post-silicon validation is shown in Fig. 1. The validation consists of four parts: pseudo functional test generation, layout-aware path selection, post-ATPG processing for PSN, and sensitivity measurement. In the following sections, we will focus on the second and third parts.

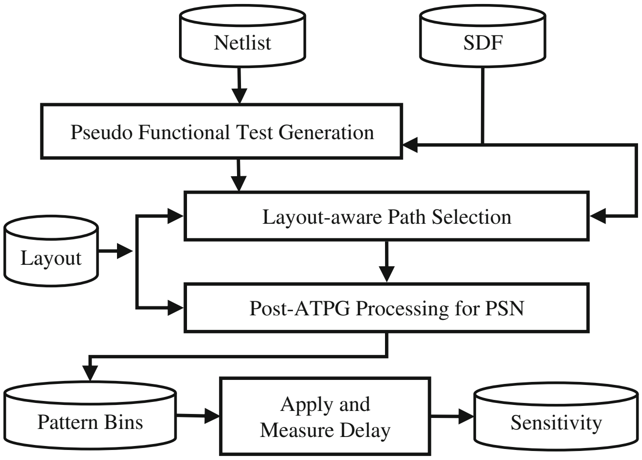

Netlist
SDF
Pseudo Functional Test Generation
Layout-aware Path Selection
Layout
Post-ATPG Processing for PSN
Pattern Bins
Apply and Measure Delay
Sensitivity

Fig. 1 Pattern generation and post-silicon validation

2.1 Pseudo Functional Test Generation

To understand timing sensitivity during functional operation, the structurally generated test patterns should mimic the functional PSN environment. The optimal solution is using functional patterns, but automatic generation of functional path delay tests is currently infeasible. Here we use a pseudo functional *K* longest path per gate (PKLPG) pattern generation tool called *CodGen* [1].

PKLPG is applied by scanning in a pattern, clocking the circuit with several medium-speed functional cycles (termed *preamble* cycles), launching the at-speed test and then scanning out the response. The preamble cycles ramp up off-chip inductor currents and minimize *dI/dt* noise. They also filter out most non-functional state transitions, so the at-speed test can be viewed as approximately functional. Correspondingly, the on-chip IR-drop is similar to functional operation.

2.2 Layout-Aware Path Selection

In order to obtain the timing sensitivity to PSN, the chip layout must be considered during the path generation process. The goal is to select paths so that the power delivery network can be characterized. Actually PKLPG can be viewed as a layout-aware path generation engine since it regards each gate as a fault site, i.e. it targets every node connected to the power delivery network.

However, it is not necessary to select every path of the circuit for sensitivity analysis. If all paths through a gate have large timing slack, PSN-induced delay can never cause these paths to fail. For that small region where the gate is located, we do not need to extract the timing sensitivity information. In contrast, in regions where low-slack (long) paths are clustered, enough paths must be generated to characterize the power delivery network.

Therefore, we can divide the circuit into small regions and weight each region using static delay information. For regions containing gates with less timing slack, we assign large K to the gates. For other regions, we may have smaller K or do not target those gates. Here, a region is a small area in the layout, inside which the sensitivity of each gate is similar. Gates inside a region will have the same K since any one of the gates can be used to characterize the region. In this paper, we assume $K=1$ (one rising and one falling path through the fault site) is good enough for PSN characterization.

After the paths are generated, a group of paths are selected for characterization. We use two basic rules to select a path: (1) the length of the path is among the top 30 %; and (2) the path covers regions not covered by previously chosen paths. Prior experience shows that the real critical paths will usually be in the top 30 % from static timing analysis. Each region spans only one row in the layout and it has a much smaller range compared to the critical range in [17]. The finer granularity guarantees the assumption that gates inside each region have similar sensitivity. In Fig. 2, an example layout is given and divided into a 3×3 grid. Each region is indexed with X and Y values. The Y value is simply the row number in the layout. The X value is the index of the power/ground segment.

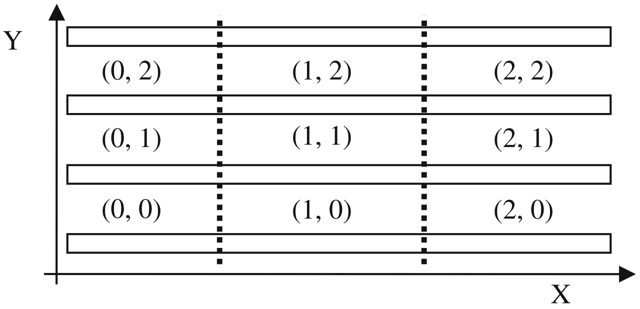


Fig. 2 An example of 3×3 layout regions

Theoretically, we only need one path to cover each region we care about. In our experiments, each region is covered up to 10 times to achieve higher accuracy. The path selection algorithm is shown below.

```
Algorithm 1: Path-Selection
  1) Read in circuit and layout
  2) Divide layout to regions and assign
     region ID to gates
  3) Read in all PKLPG paths and rank them by
     length
  4) For each region ($X_i$, $Y_j$)
     a) Initialize number of paths covering
        it as Cover ($X_i$, $Y_j$) = 0
  5) For each path $P_n$ among the top 30 %
     a) Cover_New_Region = 0
     b) For each on-path gate $G_i$ with region
        ID ($X_i$, $Y_i$)
        i) If Cover ($X_i$, $Y_i$) < 10 and not
           covered by $P_n$
             Label the region covered by $P_n$
             Cover_New_Region ++;
     c) If Cover_New_Region > 0
        Add $P_n$ to the selected path list
```

The complexity of Algorithm 1 is linear in the size of the circuit. Since we look at the K longest path through each gate, the number of paths in top 30 % is linear in the circuit size. For each long path, the algorithm iterates over its on-path gates, the number of which can be bounded by a constant. Therefore, the complexity is $O(K{\cdot}L{\cdot}N)$, where L is the length of the longest testable path and N is the circuit size.

2.3 Post ATPG Processing for PSN

Bit-Flip is a simulation-based PSN control algorithm [17]. It starts by fetching a test pattern and the corresponding path. *Critical cells*, which are neighboring-row gates located within a certain distance (*critical range*) of on-path gates, are identified by layout analysis. Unspecified or don't care (X) bits in the test pattern are initially randomly filled and the *effective weighted switching activity* (EWSA) is computed as the initial $EWSA_{max}$. EWSA is the sum of the weighted switching activity (WSA) of transitioning critical cells. Then Bit-Flip enters an iterative PSN control step and randomly flips a group of bits to improve the EWSA. In each iteration, up to G don't-care bits are randomly selected and their values flipped. Incremental logic simulation is used to compute the change in EWSA. The flip is retained if EWSA does not decrease. To reduce CPU time and direct the search, the algorithm starts with large group size G and then gradually shrinks G during the iterations.

Improved Bit-Flip (iBF) [18] combines random flipping with deterministic pattern modification to achieve similar PSN level in less CPU time. It first uses random flipping to cover most easily controlled critical cells. Then it uses background patterns as a guide to modify the test pattern in order to cover random-resistant cells. The idea is similar to a typical ATPG flow. Experimental results on benchmark circuits have demonstrated the effectiveness of these two methods.

In this work, we incorporate Bit-Flip and iBF to support pattern generation for post-silicon validation as shown in Fig. 3. The scheme can be used to increase or reduce EWSA in order to meet a specified PSN goal. This permits generating a family of patterns for a target path with different PSN levels. The control mode is set based on the current PSN level vs. the goal. If the current PSN is smaller than the goal, MAX mode is used to increase PSN. Otherwise, MIN mode reduces switching activity. The tool switches between Bit-Flip and iBF whenever 50 iterations in a row fail to increase (or

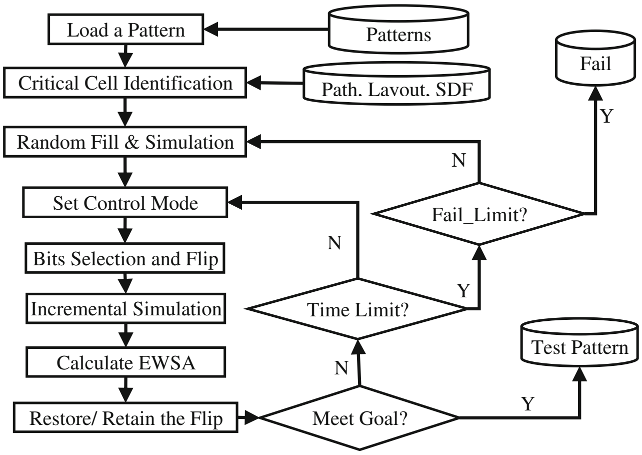


Fig. 3 Post-ATPG processing flow

decrease) PSN. If it fails to bring the PSN to the goal level within 10,000 total iterations, it will restart with a new randomly-filled pattern. The number of restarts is also limited. If the target PSN cannot be achieved after enough attempts, a failure will be reported.

2.4 PSN Estimation

Dynamic IR-drop analysis is accurate but usually requires time-consuming power-grid simulation, making it too expensive for integration into an ATPG flow. Although WSA lacks accuracy in PSN estimation, it is widely used to approximate PSN due to its low computation complexity. Many approaches have been proposed to enhance the accuracy of WSA-based PSN estimation by incorporating either spatial or temporal information, or both [14].

Simulation results in [8] demonstrated that local transitions have a relatively large impact on the voltage level of the on-path gate. They proposed to generate as many transitions on neighboring signal lines as possible in order to maximize PSN for a given path. Bit-Flip and iBF also used a similar model to select critical cells. Transition time differences between the critical cells and on-path gate is another important factor that should be considered. In the following, we will investigate the PSN impact of transition timing difference (T_d) and physical distance from the on-path gate to the critical cell.

The circuit in Fig. 4 is used to study the impact of timing and location on power supply noise. The power grid line is modeled as a RC tree (R_p and C_p are parasitic resistance and capacitance respectively). In total, N independent inverters are placed in the row ($N= 19$). Each inverter has its own input voltage source (V_{ini}) and output load (C_l). In our experiment,

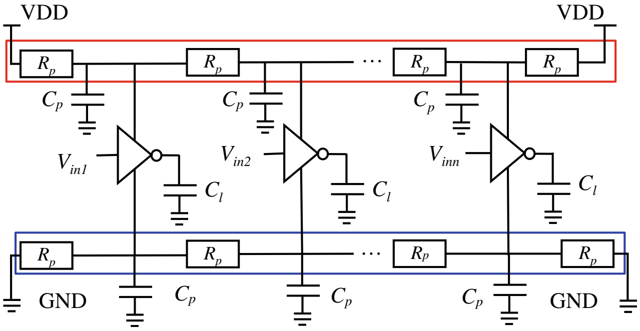
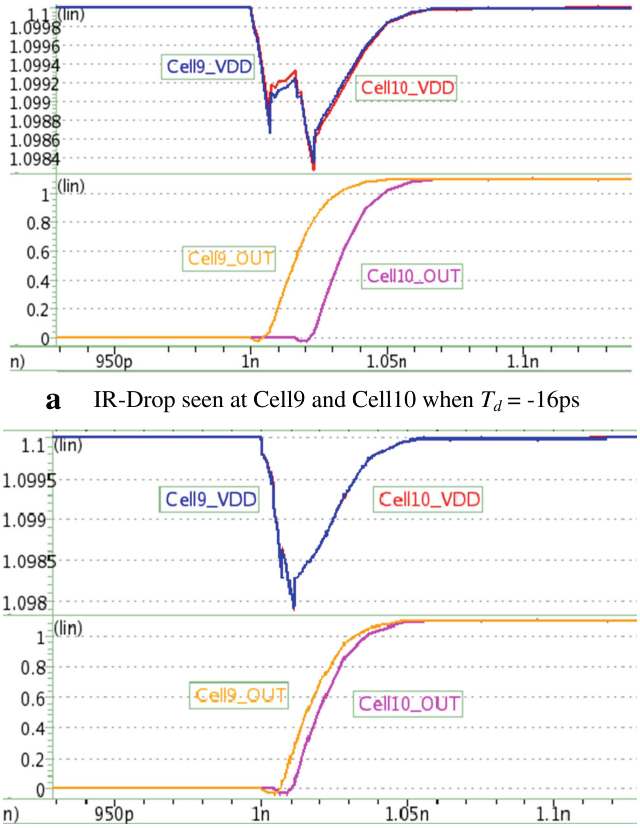


Fig. 4 Circuit to study IR-drop



a IR-Drop seen at Cell9 and Cell10 when T_d = -16ps

b IR-Drop seen at Cell9 and Cell 10 when T_d = -4ps

Fig. 5 Voltage waveforms at Cell9 and Cell10 with different transition time difference

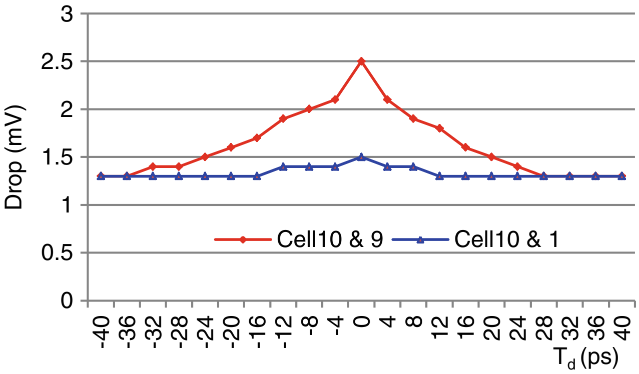


Fig. 6 Minimum voltage seen at Cell10

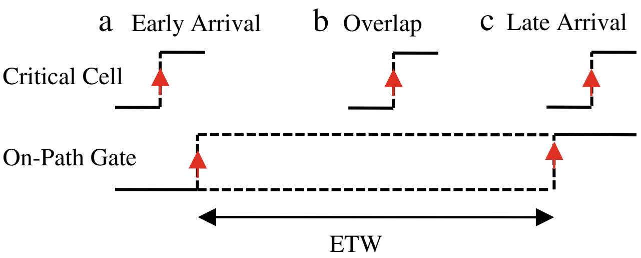


Fig. 7 Alignment of transitions on critical cell and on-path gat

ground line parasitic R_p and C_p are not used, since the focus is IR-drop on the supply. The parameters are set as C_p=0.06fF, C_l=0.5fF, and R_p=4Ω, based on the 45 nm NanGate OpenCell Library. With this circuit, we can align the transition time of different cells and study how T_d can affect the voltage level. For simplicity, we only consider how transitions on Cell1/Cell9 can affect the voltage at Cell10 which is located in the middle of the row. $T_d = T_i - T_{10}$, where i=1 or 9, and T_i is the transition time.

In Fig. 5a and b, we depict the V_{DD} level (upper half of Fig. 5a and b) and the output transitions (lower half of Fig. 5a and b) of Cell9 and Cell10 when the Cell9 transition arrives 16 ps earlier (T_d=−16 ps) and 4 ps earlier (T_d=−4 ps) than Cell10. The difference of the minimum voltage at Cell10 is 0.4 mV. In Fig. 5a, the transition of Cell9 causes a drop at Cell10 and reaches the first trough after several picoseconds. Then the voltage starts to recover. When Cell10 begins to transition, the voltage seen by Cell10 has already ramped up. Therefore, the impact resulting from Cell9's transition is reduced. With larger T_d, the IR drop caused by the two transitions will be separated from each other (non-overlap). In Fig. 5b, Cell10 switches while the drop caused by Cell9 has not recovered. So Cell10 will see a larger drop.

In Fig. 6, we sweep T_d from −40 ps to 40 ps and plot the minimal voltage at Cell10. There are two groups of experiments: (1) Cell10 and Cell9; and (2) Cell10 and Cell1. The voltage drop is maximized for small $|T_d|$. The difference between the MAX and MIN T_d, at which the other gate can affect the voltage of the on-path gate, is called the *effective timing window (ETW)*. Gates located far away have a small ETW and relatively smaller impact on the IR-drop amplitude with the same T_d. In this experiment, Cell9 caused 1 mV higher IR-drop than Cell1. These observations demonstrate the necessity of considering timing information in PSN control.

As in Bit-Flip, critical range is used to identify the small region around the on-path gate. Gates located in this region are critical cell candidates. The next step is to filter out the gates located outside the effective timing window. As shown in Fig. 7, we only consider critical cell transitions arriving within the timing window as in (b). If the output transition of a cell arrives too early (a) or too late (c), it is not considered a critical cell.

For cells with transitions that fall within the timing window, a weight based on its distance to the on-path gate is assigned to quantize the significance of its impact on the on-path gate. The weight is calculated using the formulas below, termed *critical WSA* (CWSA). If a critical cell is critical to multiple on-path gates, the maximum weight will be assigned.

$$WSA_i = \begin{cases} 1, & if\ FO = 1 \\ FO + 1, & otherwise \end{cases}$$

$$W_i = \frac{1}{1 + D_i / W}$$

$$CWSA_i = W_i {*}\ WSA_i$$

where FO is the number of fanouts, D_i is the distance between critical cells and on-path gate, and W is the critical range.

```
Algorithm 2: Critical-Cell-Identification
 1) For each on-path gate $G_i$ located at
    ($X_i$, $Y_i$) with delay $d_i$
     a) For each Gate $G_j$ located in row $Y_i$
        i) If $G_j$ located within[ $X_i$ - R, $X_i$+R)
           and $d_i$ fall within [ $d_i$ - ETW/2,
           $d_i$+ETW/2)
             Calculate the new CWSA$_j$
             If new CWSA$_j$ is larger than
             current
               Update to new CWSA$_j$
     b) If $Y_i$ - 1>= 0
           Repeat step a) for row $Y_i$ - 1
     c) If $Y_i$+1<= MAX_ROW_INDEX
           Repeat step a) for row $Y_i$+1
```

As discussed above, the critical cell identification and CWSA update can be done by Algorithm 2. In this algorithm,

Table 1 Regions and critical cells per path

Circuit	Regions (XxY)	# Covered Regions	Avg. # CC/Path		
			W/o Filtering	W/ Filtering	Filtered (%)
s35932	23x118	2380	2132	508	76.17
s38417	24x123	811	4509	910	79.82
s38584	24x122	735	2299	490	78.69

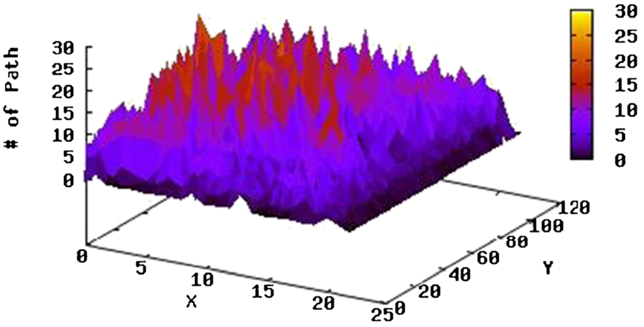


Fig. 8 Number of paths covering each region in s35932 (top 30 %)

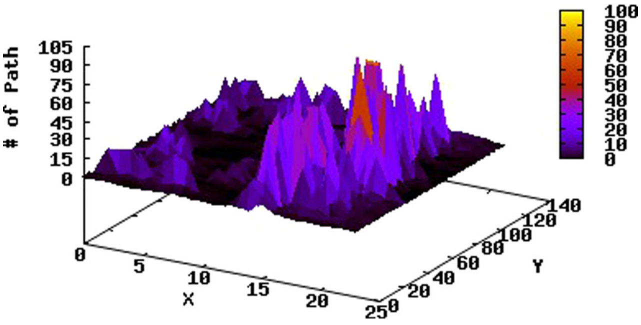


Fig. 10 Number of paths covering each region in s38417 (top 60 %)

R is the horizontal distance from the current on-path gate (Critical range in [17]).

3 Exeperimental Results

The proposed pattern generation scheme is validated on the three largest ISCAS89 benchmark circuits (s35932, s38417 and s38584), running on a 3.18 GHz Dell Optiplex 960 with 4 GB memory. The paths are generated using CodGen with $K=1$, launch-on-capture, one path per pattern. The number of preamble cycles is 6, i.e. 8 cycles including the launch and capture cycles. The ETW is set to±1 gate delay, based on the results in the previous section.

Table 1 lists the number of regions of each circuit, number of covered regions and the average number of critical cells per path without and with timing filtering. For circuit s35932, 87.69 % of the regions are covered by at least one critical path in the top 30 %, while the other two circuits have only have around 25 % regions covered. The last column in Table 1 shows that more than 75 % of critical cells are filtered out for each circuit.

Figures 8 and 9 show the number of paths in top 30 % covering each region of s35932 and s38417, respectively. For s38417, we can see some "hot" regions which are covered by relatively large numbers of critical paths, while most of regions in s35932 see similar number of critical paths. These two circuits represent two different scenarios of path length distribution. In s38417, the long paths are clustered in a few regions, while in s35932, long paths are distributed evenly across the chip. When the paths are selected from the top 60 % for s38417, the number of regions expands considerably, as shown in Fig. 10. However, as analyzed in section 2.2, for the regions which do not have critical paths going through them, there is no need to characterize the timing sensitivity, since short path never fail due the delay caused by PSN. Also notice that, although we allow each region to be covered by up to 10 paths, there are regions with more than 10 paths. The reason is that paths that cover regions with fewer than 10 paths go through regions that are already covered by 10 or more paths.

The PSN level seen by each path is computed as:

$$P = \sum_i CWSA_i / \sum_j CWSA_j$$

where *i* is a transitioning critical cell and *j* is a critical cell. In Fig. 11, we present the PSN control results for s35932. The target PSN level is set to 10, 20, 30 and 40 % and patterns generated for each long path. As shown in the figure, most paths achieve close to their target PSN. For timing sensitivity analysis, we can apply these patterns from low to high noise level and measure the FMAX. In Fig. 12, we give the average

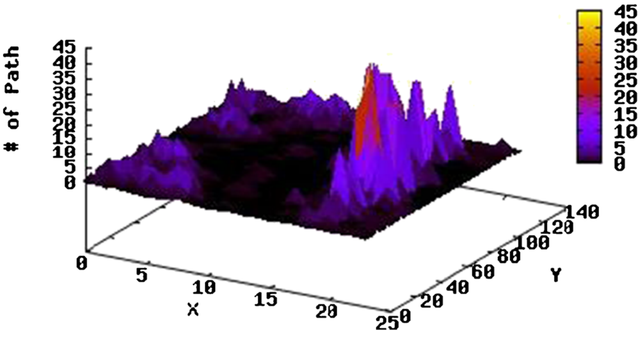


Fig. 9 Number of paths covering each region in s38417 (top 30 %)

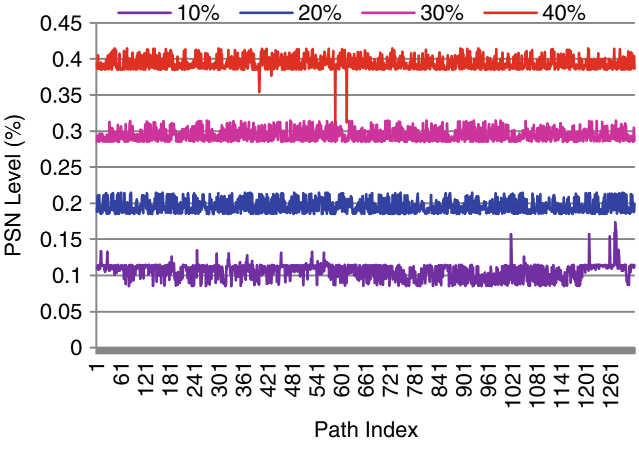


Fig. 11 s35932: Test pattern with different PSN level

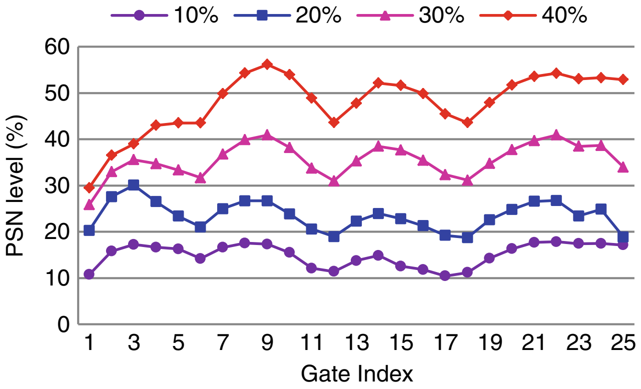


Fig. 12 PSN level of each on-path gate

PSN level of each on-path gate, indexed by its location on the path. The on-path gate PSN level is computed similar to the critical path PSN level. For each on-path gate, we have a list of its critical cells. Therefore, after the PSN control commits, the PSN level per gate can be computed as the CWSA sum of its transitioning critical cell over the CWSA sum of all of its critical cells. We can see that for this circuit, the PSN level of each on-path gate is higher than the overall PSN level.

If the PSN achieved by the algorithm falls within±1.5 % of the target level, we will report success. Otherwise, a fail is reported. Some of the patterns are intrinsically noisy, which means the pattern itself causes PSN higher than the upper bound. It is impossible to bring down the PSN level. The intrinsic PSN can be calculated by simulating the partially specified pattern. Such patterns are termed a noisy pattern (NP).

The detailed pattern generation results are listed in Table 2. In columns 1 and 2, we list the circuit and number of paths selected (#P). Column 3 lists the target PSN. Column 4 lists the number of NP. In column 5, we list the number of initial failures (after the first try), the number of patterns that succeeded after restart (#Res), and the number of failed patterns (#Fail). Column 6 lists the number of successfully filled patterns (#Suc). Column 7 lists the CPU time cost for post-silicon PSN control. The number of paths selected for characterization is at most 10 times the number of regions in the layout, since we allow up to 10 paths covering each region. Here we can see that the number of selected paths is much smaller, even for s35932, where up to 87.69 % of its regions are covered. The reason is that each path will typically cover multiple regions. In Table 3, we listed the experimental results of targeting 20 % PSN level without critical cell filtering. The experimental result on s35932 and s38584 clearly shows the benifit of critical cell filtering - reducing CPU time cost. The higher CPU time cost on s38417 is caused by the large number of inital fails.

Table 2 PSN post processing result w/ filtering

Circuit	#P	PSN	#NP	#InitFail (#Res/#Fail)	#Suc	Time (s)
s35932	1320	10 %	9	62 (25/37)	1274	1037
		20 %	0	0 (0/0)	1320	143
		30 %	0	0 (0/0)	1320	152
		40 %	0	5(0/5)	1315	264
s38417	183	10 %	178	2(0/0)	3	39
		20 %	46	19(4/15)	122	212
		30 %	1	12(1/11)	171	216
		40 %	0	100(7/93)	90	2011
s38584	245	10 %	72	22(4/18)	155	282
		20 %	6	7(1/6)	233	123
		30 %	0	42(6/36)	209	617
		40 %	0	139(13/126)	119	2103

Table 3 PSN post processing result w/o filtering

Circuit	PSN	#NP	#InitFail (#Res/#Fail)	#Suc	Time (s)
s35932	20 %	0	0 (0/0)	1320	364
s38417	20 %	0	5(1/4)	179	190
s38584	20 %	0	8(4/4)	241	316

As discussed in the previous section, the timing information will filter out gates that do not affect the delay of the path. To further understand how this affects the number of critical cells per gate, we depict the average number of critical cells per on-path gate in Fig. 13. The on-path gates are indexed in the order they appear along the path. The results clearly show that the first ~20 gates have a relatively large number of critical cells. The reason is that most of the transitions in the circuit happen too early to affect the voltage level of later gates. Therefore, the algorithm is dominated by the first ~20 gates. Without the filtering, the number of critical cells per gate is dependent on the critical range size. Since each gate has a similar

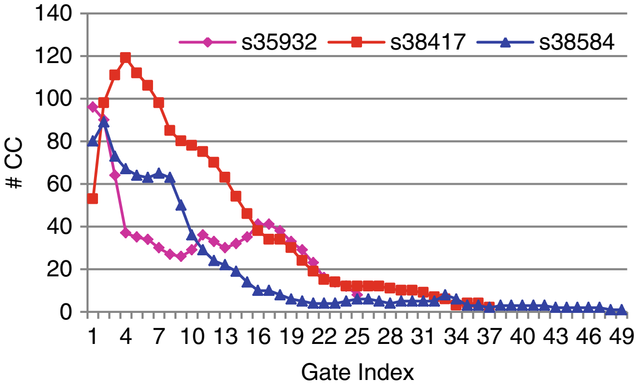


Fig. 13 Average number of critical cells per on-path gate

number of critical gates, filtering prevents simulation time from being spent on critical cells that do not affect timing.

4 Conclusion and Future Work

In this work, we addressed the problem of automatic test pattern generation for understanding the timing sensitivity to power supply noise during post-silicon validation. Experimental results demonstrate that patterns with different PSN levels can be generated.

In the future we will incorporate additional power grid hierarchy and validate the correlation between CWSA and measured IR drop. We will also apply the generated patterns to real chips and measure the delay of the paths. Then we can extract sensitivity information.

Acknowledgments This work was supported in part by the Semiconductor Research Corporation under contract 2010-TJ-2096 and by the National Science Foundation under grant CCF-1117982.

References

1. Bian K, Walker D M H, Khatri S, and Lahiri S (2013) Mixed structural-functional path delay test generation and compaction. Proc. IEEE Int Symp Defect Fault Tolerance VLSI Nanotechnol Syst pp. 7–12.
2. Cadence Design Systems, Voltage Storm Power Verification Datasheet, Retrieved from http://www.cadence.com/rl/Resources/datasheets/voltage_storm_ds.pdf, April 28, 2014.
3. Fan X, Reddy S, and Pomeranz I (2011) Max-fill: A method to generate high quality delay tests. Proc Int Symp Des Diagn Electr Circ Syst pp. 375–380
4. Fang L and Hsiao M S (2008) A fast approximation algorithm for MIN-ONE SAT. Proc Des Autom Test Eur pp. 1087–1090
5. Krstic A, Jiang YM, Cheng KT (2001) Pattern generation for delay testing and dynamic timing analysis considering power-supply noise effects. IEEE Trans CAD 20(3):416–425
6. Li X, Rutenbar R R, and Blanton R D (2009) Virtual probe: a statistically optimal framework for minimum-cost silicon characterization of nanoscale integrated circuits. Proc. IEEE/ACM International Conference on Computer-Aided Design pp. 433–440
7. Lin YC, Lu F, Cheng KT (2006) Pseudo functional testing. IEEE Trans CAD 25(8):1535–1546
8. Ma J, Tehranipoor M (2011) Layout-aware critical path delay test under maximum power supply noise effects. IEEE Trans CAD 30(12):1923–1934
9. Pei S, Li H, Li X (2012) A high-precision on-chip path delay measurement architecture. IEEE Trans VLSI 20(9):1565–1577
10. Remersaro S, Lin X, Zhang Z, Reddy SM, Pomeranz I and Rajski J (2006) Preferred fill: A scalable method to reduce capture power for scan based designs. Proc IEEE Int Test Conf pp. 1–10
11. Sde-Paz S and Salomon E (2008) Frequency and power correlation between at-speed scan and functional tests. Proc IEEE Int Test Conf pp. 1–9
12. Wang J, Qiu W, Fancler S, Walker D, Lu X, Yue Z.and Shi W (2005) Static compaction of delay tests considering power supply noise. Proc IEEE VLSI Test Symp pp. 235–240
13. Wen X, Yamashita Y, Morishima S, Kajihara S, Wang L-T, Saluja K K and Kinoshita K (2005) Low-capture-power test generation for scan-based at-speed testing. Proc IEEE Int Test Conf pp. 1019–1028
14. Wu M F, Pan H C, Wang T H, Huang J L, Tsai K H and Cheng W T (2010) Improved weight assignment for logic switching activity during at-speed test pattern generation. Proc. Asia and South Pacific Design Automation Conference pp. 493–498
15. Yuan F, Liu X, and Xu Q (2011) Pseudo-functional testing for small delay defects considering power supply noise effects. Proc. IEEE/ACC Int Conf Comput-Aided Des pp. 34–39
16. Zhang T, Gao Y and Walker D M H (2014) pattern generation for understanding timing sensitivity to power supply noise. Proc. IEEE North atlantic test workshop pp. 61–64
17. Zhang T and Walker D M H (2013) Power supply noise control in pseudo functional test. Proc. IEEE VLSI Test Symposium pp. 1–6.
18. Zhang T and Walker D M H (2014) Improved Power supply noise control in pseudo functional test. Proc. IEEE VLSI Test Symposium pp. 1–6
19. Zhang X, Ye J, Hu Y, and Li X (2013) Capturing post-silicon variation by layout-aware path-delay testing. Proc. Des Autom Test Eur pp. 288–291

Tengteng Zhang received his B.S. from Southeast University, Nanjing, China, in 2010. He was a research assistant in the State Key Laboratory of Computer Architecture, Institute of Computing Technology, China, from June 2010 to June 2011. He is currently a Ph.D. candidate in the Department of Computer Science and Engineering, Texas A&M University. His research interests include power-aware delay testing, functional power supply noise correlation and post-silicon validation.

Yukun Gao is an M.S. candidate in Department of Computer Science and Engineering, Texas A&M University. His research interests include memory delay testing, memory modeling, test scheduling, Yukun has received his B.S. in Electronic Engineering from Shandong University, China, in 2011. He worked as research assistant in the Center for Microwave and RF Technologies, Shanghai Jiaotong University, China from June 2011 to June 2013.

D. M. H. Walker is a professor in the Department of Computer Science and Engineering at Texas A&M University. His research interests include defect-based test, realistic fault modeling, IDDQ test, delay test, defect diagnosis, and yield modeling. Prof. Walker has a B.S. in engineering from the California Institute of Technology, and an M.S. and Ph.D. in computer science from Carnegie Mellon University. He is a member of the IEEE, the ACM, and Sigma Xi.