Power Droop Testing

Ilia Polian and Alejandro Czutro
Albert-Ludwigs University of Freiburg

Bernd Becker
Albert-Ludwigs University of Freiburg

Sandip Kundu
University of Massachusetts, Amherst

Editor's note:
Power droop can cause an IC to fail, but such failures cannot currently be screened during testing, because they are not covered by conventional fault models. The authors propose a heuristic method to generate test sequences that create worst-case power droop by accumulating high- and low-frequency effects.

—Kenneth M. Butler, Texas Instruments

STATE-OF-THE-ART high-performance digital ICs manufactured in deep-submicron technologies tend to draw considerable amounts of power during operation. Large transients—that is, sharp changes in power consumption—can occur within a few clock cycles. One example is a microprocessor in the idle mode that must start complex calculations that use multiple fixed-point and floating-point units simultaneously. The difference between power consumption in idle mode and at peak usage can exceed 100 W. The transition can take around 1 ns on a multi-GHz machine.

Power droop describes the impact of power consumption transients on the logic values of a circuit's signal lines and, ultimately, on the correctness of the circuit's operation.[1–3] It is related, yet not identical, to static *IR* drop and ground bounce. Although power droop could cause an IC to fail, such failures cannot be screened during testing, because conventional fault models do not cover them. In this article, we present a technique for screening such failures. We propose a heuristic method to generate test sequences that create worst-case power drop by accumulating high- and low-frequency effects. We employ a dynamically constrained version of the classical D-algorithm, which generates new constraints on the fly, for test generation. The obtained patterns can be used for manufacturing test and early silicon validation. We have implemented a prototype ATPG to demonstrate the feasibility of this approach.

Low- and high-frequency power droop

We distinguish between low- and high-frequency power droop. Low-frequency power droop (LFPD) occurs when the voltage regulator module (VRM) cannot handle large transients in the global power consumed by the device. High-frequency power droop (HFPD) occurs when the on-chip power distribution network (power grid) cannot deliver power to an individual cell quickly enough.

Figure 1 illustrates LFPD. The figure shows the circuit under test (CUT) connected to the VRM. L denotes the parasitic inductance of the interconnect. We call a sudden increase in current I demanded per unit time t a dI/dt event. After a dI/dt event, the CUT's power supply voltage V_{DD} will decrease by $L(dI/dt)$. For a current transient of 100 A (which translates into a power transient of 100 W for $V_{DD} \approx 1$ V) occurring within 10^{-9} seconds, or three cycles on a 3.3-GHz machine, this value is dramatic even for inductances L far below 1 nH.

Adding capacitance C, as shown in Figure 1, mitigates the voltage drop to cover the CUT's short-term demand for current after a dI/dt event. However, if the capacitor discharges before the VRM is ready to

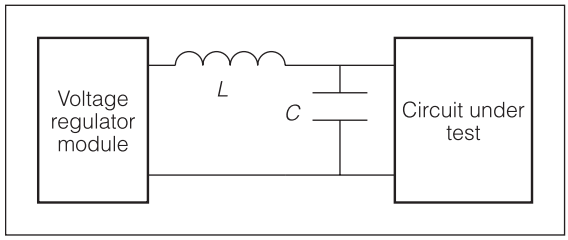

Figure 1. Circuit under test connected to a voltage regulator module, including capacitance *C* and the interconnect's parasitic inductance *L*.

0740-7475/07/$25.00 © 2007 IEEE Copublished by the IEEE CS and the IEEE CASS

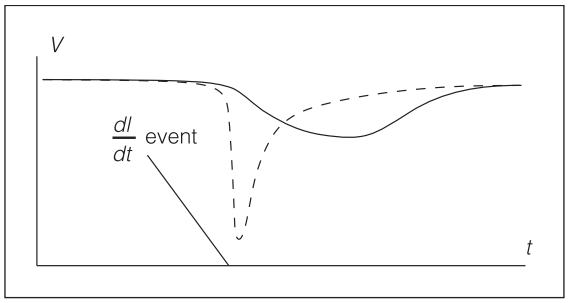

Figure 2. Voltage on the circuit under test after a *dI/dt* event, without capacitor (dashed curve) and with capacitor (solid curve).

supply the full amount of needed current, a V_{DD} drop occurs, albeit smaller and some time after the initial *dI/dt* event. As Figure 2 shows, V_{DD} declines more slowly, and after some time the VRM provides enough current and starts charging the capacitor again. (The curve shown is for illustration only and was not obtained by measurement or simulation. A typical measured voltage response is available elsewhere.[4]) Reduced V_{DD} (power starvation) can slow the switching time of the gates in the circuit, resulting in delay faults. Significantly, due to the presence of *C*, the impact is most severe several clock cycles after the actual *dI/dt* event.

In contrast to LFPD, which affects V_{DD} over the entire device, HFPD is highly localized: A few cells experience reduced V_{DD}, and the other cells operate normally. HFPD creates power starvation on the affected cells, potentially leading to delay faults on the lines (called victim lines) fed by those cells. HFPD is closely related to ground bounce and simultaneous switching noise.[5]

HFPD results from marginal power grid design.[3] The power grid of a state-of-the-art high-performance IC stretches over several metallization layers, connected by vias, as Figure 3 shows. The vias connecting power rails of different layers are relatively small and prone to marginal defects; hence, they are an obvious bottleneck for power delivery. A power delivery path to an individual cell can go through as many as seven vias. We call the part of the power rail located between two vias a segment. HFPD occurs when multiple cells drawing current from the same power grid segment suddenly increase their current demand. If the current cannot be provided quickly enough from other parts of the chip, power starvation results in a voltage drop.

Power droop is very hard to debug or diagnose, because it is nearly impossible to reproduce the error without specifically targeting the power droop conditions. There is often no hardware defect identifiable by failure analysis. Power droop effects could be wrongly attributed to radiation-induced soft errors. Ironically, one possible soft-error mitigation strategy is gate upsizing, which would actually increase power droop.

Proposed algorithm and procedure

We propose an ATPG algorithm that attempts to create worst-case power droop conditions. The generated sequence can serve in evaluating early silicon for design flaws such as insufficient sizing of the power grid. This information might be difficult to obtain analytically before actually manufacturing the IC. A second application of the generated test sequence occurs in manufacturing test. Because power droop belongs to the class of circuit marginalities,[6] the effect can be stronger on some ICs than on others. Applying the generated sequence identifies the ICs that are vulnerable to power droop so that they can be rejected or binned as low-performance parts. The patterns can be used in the presence or absence of special on-die droop detectors.[2]

The proposed ATPG procedure produces a test sequence that maximizes the effects of both LFPD and HFPD. A subsequence leading to a large *dI/dt* event and thus LFPD is generated first. As noted earlier, the LFPD will be most severe (the V_{DD} supplied to the circuit will be minimal) several clock cycles after the *dI/dt* event. At this point, the ATPG procedure generates the vectors or pairs on the basis of a model of the physical circuit, and later the ATE applies them to the actual physical circuit. The vector pair creates worst-case HFPD, resulting in a V_{DD} reduction on a victim line in addition to the global V_{DD} decrease due to LFPD. If the circuit is vulnerable to power droop, a delay fault will occur on the victim line, and this fault is propagated to an observable point. The need for the subsequence for LFPD prohibits using scan for any test vector of the sequence except the first one. Therefore, sequential test generation is required. The algorithm we propose in this article is based on constrained sequential test generation. The constraints employed maximize LFPD and HFPD; some constraints are added to the ATPG instance during the algorithm's execution.

Test generation problem

The test sequence generated for worst-case power droop creates a global *dI/dt* event stretching over

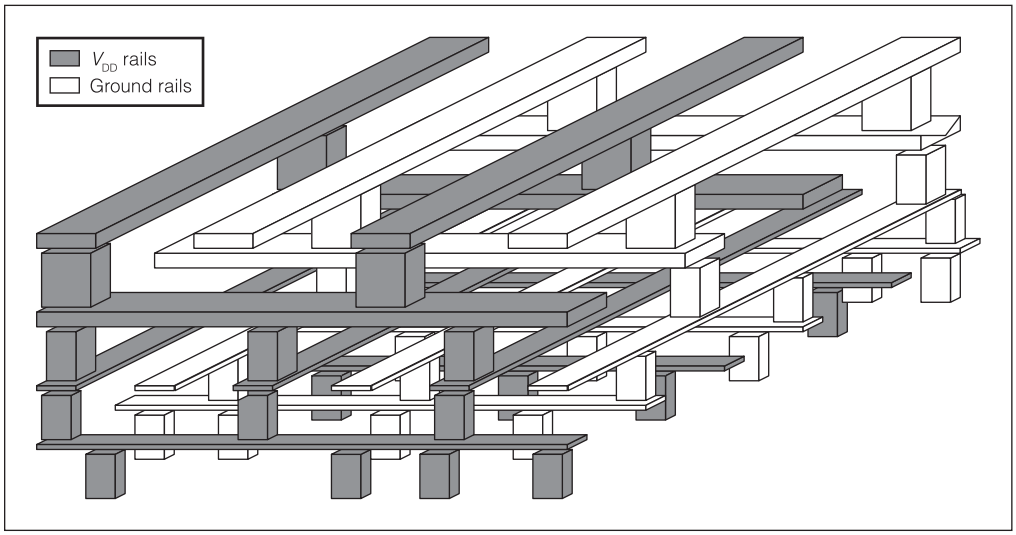

Figure 3. Four-layer power grid, with V_{DD} rails shaded and ground rails appearing as white.

multiple cycles to induce LFPD. It then imposes HFPD on a victim line v.

Creating a global *dI/dt* event requires the amount of current drawn by the circuit from the VRM to change rapidly. We do this by controlling the switching activity during application of the sequence. The sequence comprises two subsequences, $l_1l_2 \ldots l_M$ and $h_1h_2 \ldots h_N$. Subsequence $l_1l_2 \ldots l_M$ should minimize switching activity (the number of switching events) in the circuit, whereas subsequence $h_1h_2 \ldots h_N$ should maximize the switching activity (peak *n*-cycle power in Hsiao, Rudnick, and Patel's classification[7]) in the circuit.[8,9] In general, switching events on different lines consume different amounts of power. We can model this by weighting the switching activity on a node—for example, by using the load it drives as a weight. We don't currently employ such weighting, but it would integrate easily into our framework. In general, it's not possible for all the lines in the circuit to switch. For instance, if both inputs of an XOR gate have a switching event, its output cannot have one.

Worst-case HFPD occurs when the victim line v and all the aggressor lines a_1, a_2, … switch in the same direction. Aggressor lines are driven by logic cells drawing power from the same segment of the power grid as the cell driving v. Then, a significant amount of current must travel to or from a single segment of the power grid through a series of resistive and inductive vias. We can't generally impose such transitions on all the aggressors simultaneously, because of logic implications between them; hence, we require a possibly large number of aggressors to switch in the same direction as the victim. This differs from testing for capacitive crosstalk, which requires opposite transitions on the aggressor lines. HFPD leads to a delay fault on v that must be propagated to an observable point. Consequently, testing the HFPD requires a test pair (t_1, t_2), which detects the transition fault on v with additional constraints on aggressors. Mitra et al. recently published an alternative approach to test generation for HFPD but did not consider LFPD.[3]

Here, we formalize the ATPG problem for power droop for a full-scan sequential circuit. Extension to combinational circuits is straightforward. We assume that the transition on the victim line is rising. The problem formulation for the falling transition is symmetric.

The algorithm's input is as follows:

1. Circuit netlist on gate level.
2. Victim line v and list of aggressor lines a_1, a_2, … for HFPD.
3. Length of low- and high-switching-activity subsequences (LSAS and HSAS), denoted as M and N, respectively, for LFPD.

The algorithm's output is initial state s_0 of the circuit and a sequence of $M+N$ input vectors $l_1l_2 \ldots l_{M-1}h_1h_2 \ldots h_{N-1}t_1t_2$ for power droop detection, with the following constraints:

1. Line v has logic value 0 in time frame $M+N-1$ when vector t_1 is applied.
2. A stuck-at-0 fault at line v is detected in time frame $M+N$, under vector t_2.
3. As many aggressor lines a_i as possible assume logic value 0 under vector t_1 and logic value 1 under vector t_2.
4. Switching activity is as low as possible during application of the first M vectors $l_1l_2 \ldots l_{M-1}h_1$.
5. Switching activity is as high as possible during application of N vectors $h_1h_2 \ldots h_{N-1}t_1$.

The tester applies the obtained test sequence as follows: First, using scan, the initial state is shifted into the circuit's flip-flops, and the input sequence is applied to its primary inputs. We can't use scan within the sequence, because this would violate switching-activity constraints 4 and 5. Finally, the primary output is read out and the state of the circuit is scanned out. Note that the LSAS, the HSAS, and the test pair for HFPD overlap: Vector h_1 belongs to both LSAS and HSAS, and vector t_1 belongs to both HSAS and the test pair.

Constraints 1 and 2 ensure detection of the delay imposed by power droop on the victim line (rising transition fault on v). Constraint 3 creates worst-case HFPD. Constraints 4 and 5 help to induce the largest possible *dI/dt* event required for worst-case LFPD. Fault detection occurs when the combined effects of LFPD and HFPD impact the delay on v. Although satisfaction of constraints 1 and 2 is mandatory, the other constraints demand only satisfaction in as many cases as possible. These constraints can also be contradictory: An assignment necessary to satisfy constraint 5 might prevent the logic values on aggressor lines desired by constraint 3. Consequently, the problem might have multiple solutions with different degrees of satisfaction for constraints 3 through 5.

Design features present in some circuits simplify minimizing average switching activity for designs using clock gating or clock frequency control. For instance, if the clock can be switched off completely, doing so for M cycles would yield a high-quality LSAS, and no specific test generation would be needed for this part of the sequence. For scan designs, average switching activity can be low during low-speed shift and the interval between shift and launch. In this case, the test sequence might start with HSAS applied at-speed—as, for example, with transition test patterns. In this work, we assume that clocks cannot be manipulated. If the circuit contains logic BIST blocks known to have high switching activity, the ATPG could be constrained to switch them on during the HSAS. We did not use circuits with logic BIST in our experiments.

We could extend the problem formulation to multiple victim lines. However, the sets of constraints for different lines would generally not be compatible, thus reducing the solution quality for the individual lines. Because the overall number of lines susceptible to power droop is reportedly very low (fewer than 100 for a microprocessor of 128,000 standard cells[1]), applying several sequences, each addressing one line, appears to be a better strategy.

ATPG algorithm

Applying the classical D-algorithm modified to satisfy constraints 1 through 5 on an $(M + N)$ time frame expansion of the circuit solves the sequential ATPG problem. Although constraints 1 and 2 are mandatory, it's preferable for constraints 3 through 5 to hold at as many lines as possible, but we cannot expect them to always hold.

Figure 4 depicts a problem instance with all desired constraints satisfied: v is the victim, and a_1 through a_4 are the four aggressors; the other lines are not involved in HFPD. During LSAS, no line is switching; during HSAS, all lines are switching. Upon application of the test pair, the victim and all the aggressors have a rising transition. The values on the other lines are irrelevant.

To keep the problem tractable, we enforce mandatory constraints by adding them to the D-algorithm's assignment queue. To satisfy as many of the desired constraints as possible, we introduce the following three rules:

- *Rule 1:* Assume a rising transition on victim line v. When deciding on an aggressor line a_i in time frame $M + N - 1$ (under vector t_1), always assign logic value 0 first. In time frame $M + N$ (under vector t_2), always assign logic value 1 first.
- *Rule 2:* When selecting which line to make a decision on, it is better to select lines in later time frames.
- *Rule 3:* Suppose the decision has been made on line n in time frame k, $M \leq k \leq M + N - 2$, that is, under one of the vectors in HSAS: $h_1, \ldots h_{N-1}$. If line n is already assigned in time frame $k + 1$ to a logic value, assign it to the opposite logic value. Under one of the vectors in LSAS, assign it to the same logic value.

The rationale in Rule 1 is to create enough simultaneous transitions on aggressor lines to create worst-case HFPD. The purpose of Rule 2 is to facilitate the application of Rule 3. Rule 3 minimizes (maximizes) switching activity in LSAS (HSAS) to impose worst-case LFPD. If switching activity were weighted, Rule 2 could be modified to try the lines with the greatest weight first. Rule 3 is implemented by generating desired constraints on the fly: After an assignment, the ATPG algorithm generates a desired

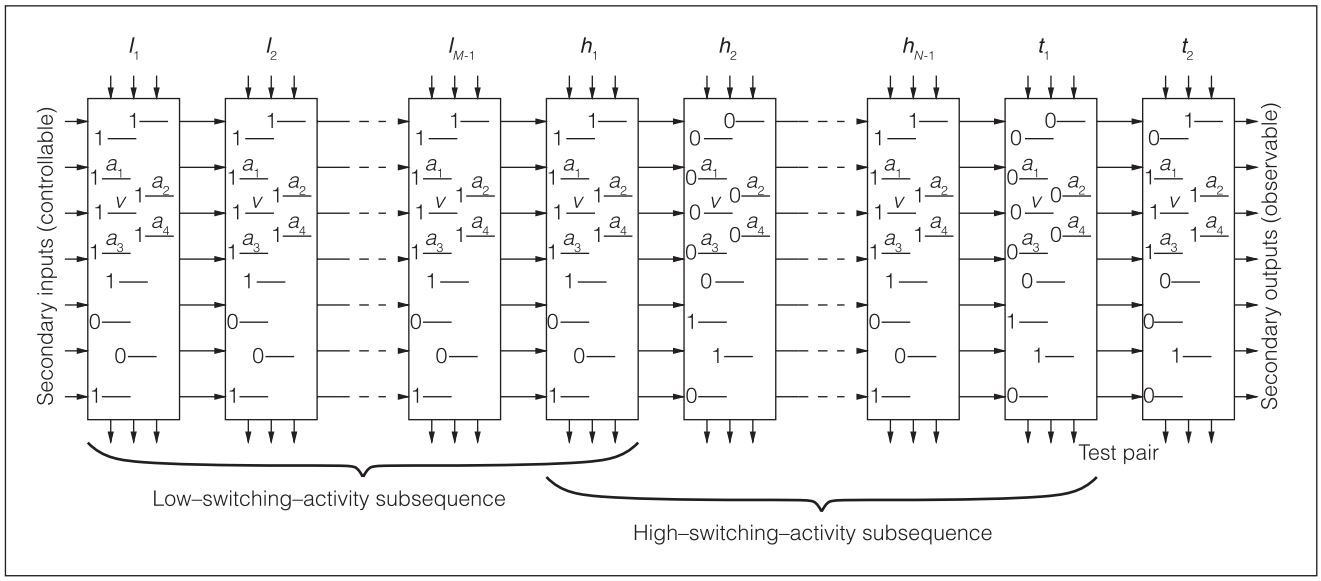

Figure 4. Time frame expansion of the circuit for power droop ATPG.

constraint for the preceding time frame. We call the resulting algorithm the dynamically constrained D-algorithm.

Complexity issues

Because test generation is an NP-complete problem, running deterministic ATPG on time-frame-expanded circuits might be impractical for realistic circuits and when the number of time frames exceeds 100. The additional constraints appear to further increase the complexity. However, most of the additional constraints are desirable, and violating them doesn't render a solution invalid. Even the subproblem of generating a sequence with worst-case switching activity hasn't been solved optimally for nontrivial circuits.[10]

It's possible to replace one ATPG run on the complete time frame expansion with a series of ATPG invocations on smaller subcircuits of the time frame expansion. The first invocation of the ATPG generates the test pair (time frames $M + N - 1$ and $M + N$), assuming that the state s_{M+N-2} of flip-flops before time frame $M+N-1$ is controllable and considering all the relevant constraints. Then, ATPG runs on time frame $M + N - 2$ using the previously generated state s_{M+N-2} as an output constraint and satisfying the power droop constraints. This iterates until reaching time frame 1.

This procedure can result in reduced solution quality. On the other hand, the first ATPG invocation has essentially the same complexity as transition fault ATPG, and subsequent invocations have even less complexity. Hence, power droop ATPG using this approach is feasible for all circuits for which transition fault ATPG is applicable. Considering subcircuits consisting of several time frames can improve solution quality.

Experimental results

We applied the dynamically constrained D-algorithm for power droop test sequence generation to 1985 and 1989 IEEE International Symposium on Circuits and Systems (ISCAS) circuits. Determining the aggressor and victim lines and the lengths of LSAS and HSAS is generally beyond the scope of this article and requires layout, power grid, and technology information not available for ISCAS circuits. Victim line v can be obtained using the analysis described by Tirumurti et al.[1] Cells powered by the same segment as v drive the aggressor lines. The choice of M and N should maximize LFPD. We could have derived M and N analytically from the electrical parameters of the circuit, the VRM, the capacitor, and so on, or obtained them by measurement.

Rather than using these options, we selected as v a line with maximum fan-out, because it was likely to have the largest load, and we selected five random lines in the circuit as aggressors. We repeated the experiment using aggressors selected on the basis of their proximity to v and obtained similar results. We generated results for both the rising and the falling transition on v and different lengths of LSAS and HSAS.

Table 1 shows the results for circuits up to s09234; LSAS and HSAS lengths are 10 clock cycles. The

Table 1. Experimental results for rising transition and $M = N = 10$ clock cycles (20 time frames). (LSAS: low-switching-activity subsequences; HSAS: high-switching-activity subsequences.)

ISCAS circuit	No. of switching aggressors	Switching activity			CPU time (hr:min:s)
		LSAS (%)	HSAS (%)	Random vectors (%)	
c0017	3	0	87	47	0:00:00
c0095	3	0	71	39	0:00:00
c0880	3	0	50	34	0:00:01
c1908	4	0	57	40	0:00:01
c3540	3	0	46	33	0:00:05
c6288	3	0	47	38	0:00:39
c5315	4	0	53	40	0:00:14
c7552	5	0	52	41	0:00:42
s00027	2	11	69	37	0:00:00
s00208	1	0	36	19	0:00:00
s00298	1	10	50	36	0:00:01
s00386	2	4	30	36	0:00:02
s00382	2	4	32	34	0:00:02
s00344	3	16	47	35	0:00:02
s00349	1	3	40	35	0:00:01
s00400	2	5	37	34	0:00:02
s00444	1	0	39	31	0:00:07
s00526	1	3	24	28	0:00:01
s00510	1	0	29	24	0:00:04
s00420	2	3	26	13	0:00:02
s00832	3	0	37	30	0:00:04
s00820	2	0	43	30	0:00:01
s635	1	0	13	10	0:00:02
s00641	4	0	50	31	0:00:06
s00953	1	1	19	17	0:05:06
s00713	2	2	49	29	0:00:15
s00838	3	0	27	10	0:00:04
s938	1	0	28	10	0:00:05
s01238	1	13	36	26	0:00:23
s01196	3	18	38	27	0:00:21
s01494	1	29	34	31	0:00:32
s01488	1	9	33	31	0:00:56
s01423	1	5	26	32	0:00:46
s1512	1	0	29	24	0:00:15
s3271	5	32	55	49	0:00:55
s3384	4	23	58	46	0:01:39
s3330	3	9	52	36	0:01:06
s4863	3	24	40	42	1:34:17
s05378	2	6	53	35	0:08:38
s6669	4	18	45	38	0:06:25
s09234	4	16	41	29	0:35:08
Average	2.37	6.44	42.15	31.43	0:03:53

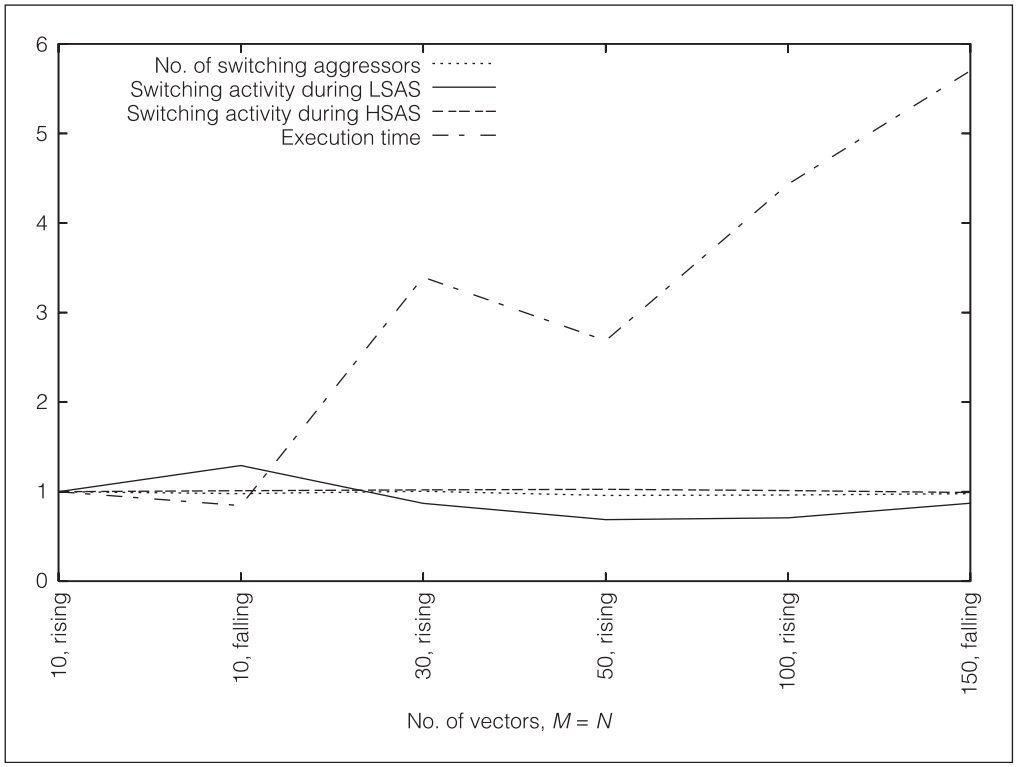

Figure 5. Number of switching aggressors, switching activity, and execution time for *M* = *N* = 10, 30, 50, 100, and 150 clock cycles, and for rising and falling transitions (normalized).

second column shows how many of the five aggressors the supporting transition can be justified on (constraint 3). Columns 3 and 4 report the average switching activity for each circuit (number of switching events per line and clock cycle) in percentage of lines having a switching event during LSAS and HSAS, respectively (constraints 4 and 5). For comparison, we determined the average switching activity of 5,000 random vectors, reported in column 5. Note that random vectors reported by Hsiao, Rudnick, and Patel achieve far higher switching activity (often more than 100%) because their results account for multiple switches within a cycle (due to glitches) and use a weighted metric.[11] Execution time on an UltraSparc-IIIi machine (2 × 1,280 MHz with 6 Gbytes of RAM) appears in the last column. Averages appear in the last row of the table.

The worst-case HFPD occurs when supporting transitions are justified on many of the aggressor lines. For approximately half of the considered circuits, the supporting transitions have been justified on three or more aggressors. The extent of the *dI/dt* event—that is, the difference between switching activity during LSAS and HSAS—reveals the severity of the LFPD. Switching activity during LSAS is 0 for combinational circuits and close to 0 for 10 sequential circuits. For 13 of the remaining 23 circuits, the switching activity during HSAS exceeds its counterpart during LSAS by a factor of 5 or more, whereas this ratio is less than 2 for only three circuits. Recall that 100% switching activity in HSAS is not generally achievable and that the largest achievable value is not even known.

Comparing switching activity during HSAS to that of random vectors, we find that the ratio is about 1.35 on average and exceeds 1.5 for 10 of the 41 circuits. Decisions previously made on LSAS and the test pair might imply assignments on HSAS that conflict with the goal of maximizing switching activity. In fact, because of such implications, switching activity during HSAS falls below that of random vectors for five circuits. Nevertheless, on average, the switching activity generated on HSAS is superior to that generated on random vectors, which don't require us to consider such implications.

We generated corresponding data for LSAS and HSAS lengths of 10, 30, 50, 100, and 150 clock cycles, and for rising and falling transitions on the victim line. Figure 5 reports only average numbers, normalized to the average data for $M = N = 10$ and the rising transition that Table 1 reports. Neither the number of switching aggressors nor the switching activity during HSAS changes significantly. Switching activity during LSAS improves (decreases) slightly for larger values of M and N. The solution quality seems to be relatively stable with respect to M and N, and we attribute some of the statistical noise to the differing random sets of aggressors throughout the experiments.

Somewhat unexpectedly, the increase in execution time is sublinear in M and N. Moreover, this increase is not monotonic. The problem's difficulty seems to depend on the specific problem instance (controllability and observability of aggressor and victim lines,

Table 2. Experimental results for rising transition and $M = N = 10$ clock cycles.

ISCAS circuit	No. of switching aggressors	Switching activity			CPU time (hr:min:s)
		LSAS (%)	HSAS (%)	Random vectors (%)	
s13207	3	9	42	27	2:18:01
s15850	3	10	34	25	1:28:18
s35932	3	25	48	42	20:02:28
s38584	2	19	46	32	6:01:27
s38417	5	12	32	26	2:01:55
Average	3.2	15.0	40.4	30.4	6:22:25

and so on) rather than simply the number of unfolded frames given by $M + N$. The line assignments imposed by the desired constraints might not be helpful for finding a valid test sequence quickly, and this dependency seems to appear quite randomly.

Table 2 summarizes the results for larger ISCAS 89 circuits. We didn't generate the complete set of results for different values of M and N for these circuits. This experiment ran on an AMD Opteron (2,600 MHz with 16 Gbytes of RAM) running Debian GNU Linux. Reported execution time might be overly pessimistic, because several other experiments were running concurrently on this computation server.

Larger ISCAS circuits are generally more challenging because there are many more flip-flops than primary inputs. Consequently, there are many reconvergencies in the unfolded circuit, meaning more decisions and possibly backtracks. Our solution quality is slightly poorer than that for smaller circuits, and the execution time is higher. Both are particularly severe for s35932. Apart from that circuit, the solution quality is largely comparable to that obtained for smaller circuits.

ALTHOUGH THE PROPOSED implementation is adequate for midsize blocks and demonstrates the feasibility of our approach, scalability might be limited for larger devices and longer test sequences. Possible solutions include the use of a better ATPG algorithm such as Podem (path-oriented decision making) in connection with advanced techniques, including static and dynamic learning. Incorporating alternative metrics for power consumption on the basis of accurate gate delay information is another possible direction for future research. ■

Acknowledgments

This work was presented at the 2006 International Conference on Computer Design. Parts of this work are supported by the German Research Foundation under grant BE 1176/14-1.

■ References

1. C. Tirumurti et al., "A Modeling Approach for Addressing Power Supply Switching Noise Related Failures of Integrated Circuits," *Proc. Design, Automation and Test in Europe* (DATE 04), IEEE CS Press, 2004, pp. 1078-1083.
2. A. Muhtaroglu, G. Taylor, and T. Rahal-Arabi, "On-Die Droop Detector for Analog Sensing of Power Supply Noise," *J. Solid-State Circuits*, vol. 39, no. 4, Apr. 2004, pp. 651-660.
3. D. Mitra et al., "Test Pattern Generation for Power Supply Droop Faults," *Proc. VLSI Design Conf.* (VLSID 06), IEEE CS Press, 2006, pp. 343-348.
4. G.H. Johnson, "Challenges of High Supply Currents During VLSI Test," *Proc. Int'l Test Conf.* (ITC 00), IEEE CS Press, 2000, pp. 1013-1020.
5. Y.-S. Chang, S.K. Gupta, and M.A. Breuer, "Test Generation for Maximizing Ground Bounce for Internal Circuitry with Reconvergent Fan-outs," *Proc. 19th IEEE VLSI Test Symp.* (VTS 01), IEEE CS Press, 2001, pp. 358-366.
6. B. Grundmann, R. Galivanche, and S. Kundu, "Circuit and Platform Design Challenges in Technologies beyond 90 nm," *Proc. Design, Automation and Test in Europe* (DATE 03), IEEE CS Press, 2003, pp. 44-47.
7. M.S. Hsiao, E.M. Rudnick, and J.H. Patel, "Peak Power Estimation of VLSI Circuits: New Peak Power Measures," *IEEE Trans. VLSI Systems*, vol. 8, no. 4, Aug. 2000, pp. 435-439.
8. A. Krstic, Y.-M. Jiang, and K.-T. Cheng, "Pattern Generation for Delay Testing and Dynamic Timing Analysis Considering Power-Supply Noise Effects," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 20, no. 3, Mar. 2001, pp. 416-425.
9. M. Nourani, M. Tehranipoor, and N. Ahmed, "Pattern Generation and Estimation for Power-Supply Noise

Analysis," *Proc. 23rd IEEE VLSI Test Symp.* (VTS 05), IEEE CS Press, 2005, pp. 439-444.

10. C.Y. Wang, K. Roy, and T.L. Chou, "Maximum Power Estimation for Sequential Circuits Using a Test Generation Based Technique," *Proc. IEEE Custom Integrated Circuits Conf.* (CICC 96), IEEE Press, 1996, pp. 229-232.
11. M.S. Hsiao, E.M. Rudnick, and J.H. Patel, "Effects of Delay Models on Peak Power Estimation of VLSI Sequential Circuits," *Proc. Int'l Conf. Computer-Aided Design* (ICCAD 97), IEEE CS Press, 1997, pp. 45-51.

Ilia Polian is a senior member of the scientific staff and chair of computer architecture at Albert-Ludwigs University of Freiburg, Germany. His research interests include defect modeling and DFT in nanoscale technologies, and formal verification of hybrid and real-time systems. Polian has a Dipl-Inf and a PhD in computer science from Albert-Ludwigs University of Freiburg. He is a member of the IEEE.

Alejandro Czutro is pursuing a Dipl-Inf at Albert-Ludwigs University of Freiburg. His research interests include compression and hardware-based decompression of test data, resistive bridging and stuck-at faults, and ATPG for signal integrity issues.

Sandip Kundu is a professor of electrical and computer engineering at the University of Massachusetts, Amherst. His research interests include VLSI circuit design, VLSI testing, CAD algorithms, and coding and information theory. Kundu has a BTech in electronics and electrical communication engineering from the Indian Institute of Technology, Kharagpur, India, and a PhD in electrical and computer engineering from the University of Iowa. He is an IEEE Fellow and an IEEE Computer Society Distinguished Visitor.

Bernd Becker is a professor on the faculty of applied sciences at Albert-Ludwigs University of Freiburg. His research interests include data structures and efficient algorithms for circuit design; design, test, and verification of circuits and systems; and multimedia in research and teaching. Becker has a Dipl-Math from the University of Saarland, Germany. He is a senior member of the IEEE.

■ Direct questions and comments about this article to Ilia Polian, Albert-Ludwigs University of Freiburg, Georges-Köhler-Allee 51, 79110 Freiburg i. Br., Germany; polian@informatik.uni-freiburg.de.