Small Delay Defect Detection Using Self-Relative Timing Bounds

Ryan Helinski, Jim Plusquellic and Mohammad Tehranipoor*

*University of Maryland, BC, *University of Connecticut*

Abstract

A novel testing strategy is proposed that is designed to detect small delay defects by creating internal signal races. The races are created by launching transitions along two paths simultaneously, a reference path and a test path. The arrival times of the transitions on a common or 'convergence' gate determine the result of the race. The presence of a small delay defect on the test path creates a static hazard on the output of the convergence gate that is directed to the input of a scan-latch. A glitch detector is added to the scan latch to record the presence or absence of the glitch.

1.0 Introduction

Advancement of technology to nanometer feature sizes and new materials is changing the pareto of defect types, making defects such as resistive open vias, mouse bites and gate oxide failures more prevalent [1]. These types of defects increase the delay of signal paths and can cause a delay fault. Those that go undetected when tested along short paths are called small delay faults. Small delay defects may cause failure when exercised along a sufficiently long path in mission mode. In either case, they are generally considered to represent a reliability problem and methods are needed to screen them effectively [2][3].

The transition fault model is widely used for delay fault test generation. Unfortunately, timing unaware transition fault ATPG does not consider path length as a criteria for generating tests and therefore targets only gross delay defects. Recent work addresses this concern by selecting test patterns that test small delay defects on long paths [4][5][6]. Other approaches target small delay defects by partitioning patterns into timing sets and testing at faster-than-at-speed [7][8][9][10][11].

In this paper, we propose a novel method for the detection of small delay defects that minimizes the number of delay tests that need to be applied at faster than the rated clock speed. The method requires the longest path to each primary output and scan-latch input (subsequently referred to as an endpoint) to be validated using delay tests or a reference path test structure (described later). We refer to these longest paths as *reference paths*. Once these paths are validated, they are used to upper bound the delay of path segments driving their off-path inputs. This process detects small delay defects that occur on the reference paths and on the path segments driving the off-path inputs.

The process of testing the off-path segments for small delay defects involves simultaneously propagating signals along both the off-path segment and reference path segment. The gate that serves as the endpoint of both path segments is called the *convergence* gate. Under nominal conditions, the off-path segment has a smaller delay or a delay equal to the reference segment delay (otherwise, the reference path is not the longest path), and its transition arrives at the convergence gate before the reference segment signal. If the off-path segment has a small delay defect, the opposite may occur. The transitions on the inputs to the convergence gate are controlled such that if the latter occurs, a glitch is produced on the output of the convergence gate. A glitch detector, placed at a point beyond the convergence gate, is used to capture the glitch, thereby recording the result of the signal race. The glitch detector value is scanned out for inspection to determine if a small delay fault occurred.

The advantages of the method include the elimination of a capture clock cycle, which significantly reduces test power issues. The method can also detect very small delay defects without the need to apply a faster-than-at-speed clock. The following sections describe the technique using an example circuit for illustrative purposes.

2.0 Concept

The proposed method makes use of internal signal races along path segments as a means of upper bounding the propagation delay of a test path segment against the delay of reference path segment. In order to accomplish this, the upper bounds on the delay of a set of reference paths are determined by applying standard delay tests or by using a reference path test structure (described later). The longest path to each endpoint (scan latch input or primary output) is selected as the reference path.

A reference path with multiple endpoints is shown in Figure 1. The longer path to endpoint *D* is labeled p_r for path$_{\text{reference}}$. A transition is launched from a PI or scan-latch *A* (subsequently referred to as a launchpoint) and is captured at endpoint *D*. If the signal propagates to *D* within the launch-capture cycle time, then its delay is upper bounded by that time. This process confirms that the reference path does not have a delay defect. The same process is applied to the remaining reference paths.

The validated reference paths are then used to bound the delays of other, shorter paths in the circuit. Figure 2 illustrates how this is accomplished. Two path segments are identified as s_r (for segment$_{\text{reference}}$) and s_t (for segment$_{\text{test}}$)

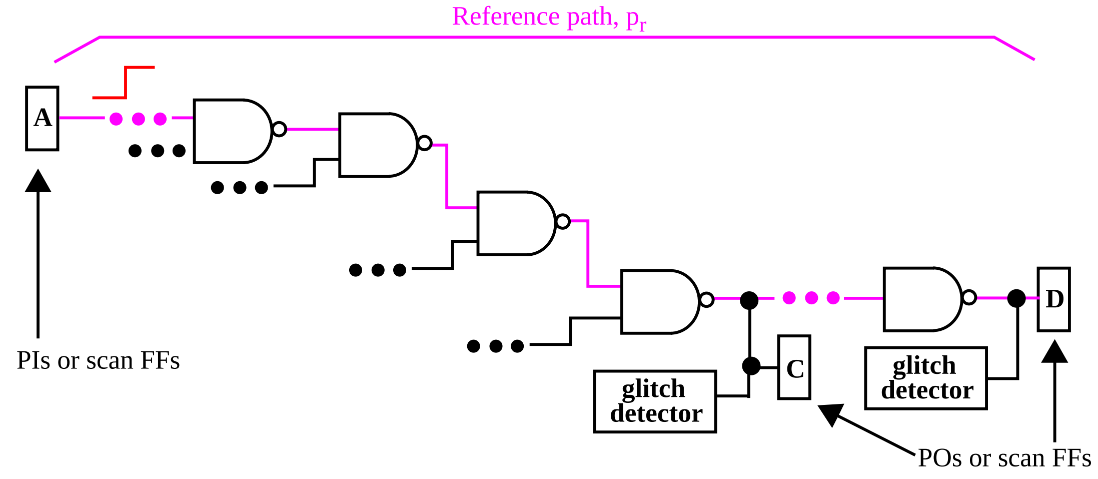

Figure 1. A reference path is the longest path to an endpoint.

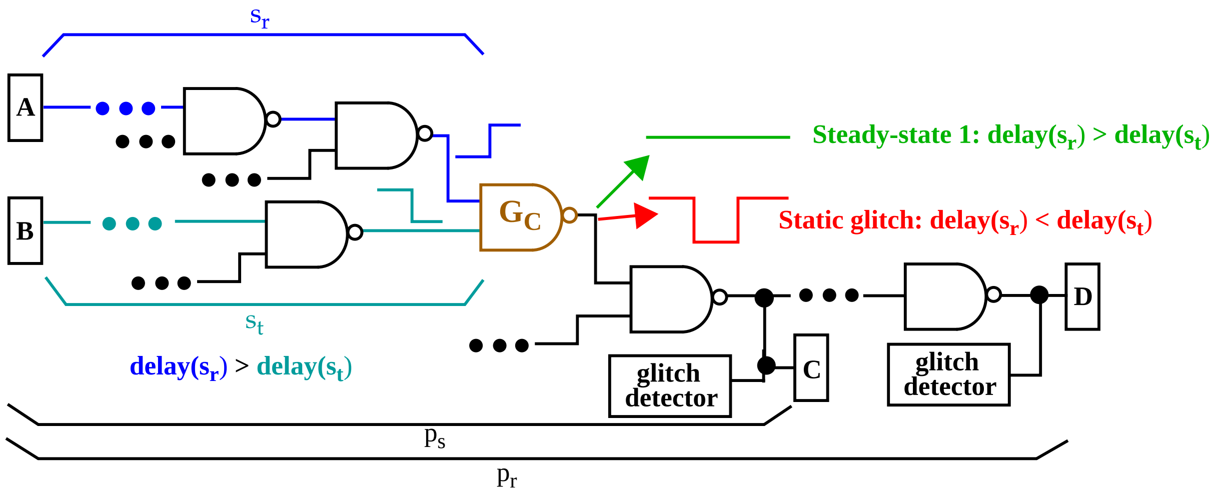
s_r
A
B
G_C
Steady-state 1: delay(s_r) > delay(s_t)
Static glitch: delay(s_r) < delay(s_t)
s_t
delay(s_r) > delay(s_t)
glitch detector
C
glitch detector
D
p_s
p_r

Figure 2. A signal race is created between a reference path segment and a test path segment driving an off-path input to a *convergence gate*.

and have launchpoints labeled *A* and *B*, respectively. The endpoints of the segments drive the inputs of a convergence gate, $\mathbf{G_C}$. Since the reference path is the longest path to endpoint *D*, the delay of s_r, delay(s_r) is greater than or equal to delay(s_t) by design. The transitions shown at the inputs of $\mathbf{G_C}$, i.e., 0 -> 1 for s_r and 1 -> 0 for s_t cause the output of $\mathbf{G_C}$ to behave in one of two ways. If delay(s_r) >= delay(s_t) then the 0 along s_t arrives before the 1 on s_r, and the output remains steady-state high. If the opposite is true, i.e., delay(s_r) < delay(s_t), then $\mathbf{G_C}$'s output switches momentarily with duration proportional to the difference in delays along the two segments.

The relative timing of the two segments is reflected in the output behavior of $\mathbf{G_C}$. One way to record the output behavior of $\mathbf{G_C}$ for subsequent inspection is to monitor the state of the path segment between the convergence gate and an endpoint using a *glitch detector.* The glitch detector is designed with a memory element that flips state if a transition occurs on its input.

Figures 1 and 2 show two glitch detectors at endpoints *C* and *D*. Although it is possible to use the glitch detector at endpoint *D* for the test shown, selecting an endpoint that is closest to the convergence gate, e.g., the glitch detector at endpoint *C*, is better for several reasons. First, differences in pull-up and pull-down strengths of gates along a path can compress the width of the glitch (and even eliminate it), making it more difficult or impossible to detect. Second, hazards produced on off-path inputs between the convergence gate and the endpoint may invalidate the result. So keeping this segment small helps minimize these effects.

The example shows the reference segment input to $\mathbf{G_c}$ changing from the dominate value (0) to the non-dominate value (1), and the test segment input transitioning in the opposite direction. Reversing these transitions allows the

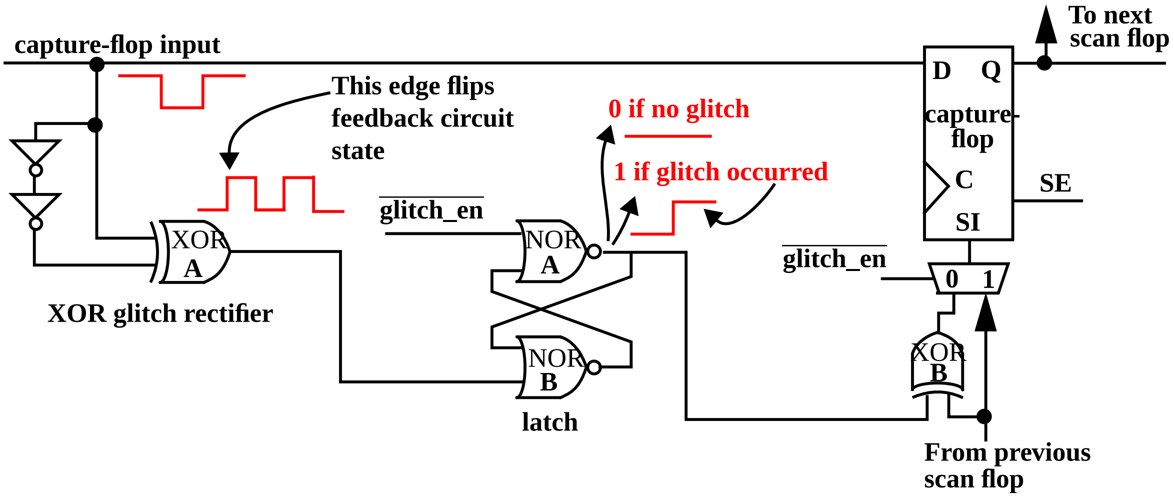

Figure 3. Glitch Detector Design

relative segment delay of the other transition along these segments to be tested. For example, assume the reference segment transition is 1 -> 0 and the test segment transition is 0 -> 1. If the reference segment transition is slower, i.e., delay(s_r) > delay(s_t) then a static hazard is produced. Therefore, the interpretation of the results is reversed for the other set of transitions on the inputs to $\mathbf{G_c}$. In this case, however, it should be noted that the reference path segment delay must be strictly larger than the test path segment delay by the inertial delay of $\mathbf{G_c}$ plus a margin because passing the test requires a glitch to be produced and this will not occur if this condition is not met.

Once a test segment has been characterized, it can then serve as a reference path for tests of path segments that converge on its off-path inputs. In this way, the method can be applied recursively, first using the longest paths as the reference paths and later using progressively shorter paths as the reference paths.

3.0 Glitch Detector

As indicated above, the width of the static hazard is proportional to the amount of additional delay introduced by a small delay defect. Therefore, the design and layout of the glitch detector needs to be optimized to detect narrow glitches in order to maximize the sensitivity of the method to small delay defects. A second design criteria is to keep it small, to minimize the overhead associated with the method. Last, the glitch detector needs to be compatible with launch-off-capture and launch-off-shift delay test methodologies in order for it to be deemed practical.

One possible implementation of a glitch detector that meets these criteria is shown in Figure 3. The *capture-flop input* is shown at the top of the figure and the *capture-flop* (with scan) is shown on the right. The remaining gates constitute elements of the glitch detector.

An *XOR glitch rectifier* is shown on the left and consists of a sequence of two inverters and an XOR gate. The inputs to *XOR glitch rectifier* are driven by the capture-flop input[1]. This circuit rectifies a negative or positive going glitch into a series of transitions on the output of XOR **A** by delaying the pair of transitions introduced by a static hazard asymmetrically to the XOR inputs. Example transitions are shown in the figure.

The XOR **A** output drives the input of a latch, i.e., two NOR gates configured with feedback. The output state of the latch is initialized to 0 prior to conducting the test by setting $\overline{glitch\_en}$ high. The $\overline{glitch\_en}$ control signal is then set low prior to application of the delay test patterns. If a static hazard is propagated to the capture-flop input as a result of the test, the first rising edge produced by XOR **A** flips the state of the latch and generates a 1 on the output of NOR **A**. Once the test result is stored in the latch, XOR **B** is used to 'insert' the result into the scan chain. This is accomplished by performing a one-bit shift of the scan chain with $\overline{glitch\_en}$ held low. If a output of the latch is 0, then the contents of the scan chain at this position remains unchanged. If the latch output is 1, XOR **B** flips the state of the bit moving into the capture-flop. The result of this test, as well as the results of other tests performed simultaneously on other paths, are scanned out after setting $\overline{glitch\_en}$ to 1.

4.0 Reference Path

The path sensitization described in Section 2.0 relies on a set of reference paths that have been validated to be free of small delay defects. The reference paths are defined as the longest paths that drive each endpoint (capture latch or primary output). Since all other paths to the endpoints are shorter by definition, it follows that identifying and validat-

[1] If the load capacitance of the inverter and XOR gate is a concern, an inverter can be inserted in series with this connection as a means of reducing the load to one inverter input.

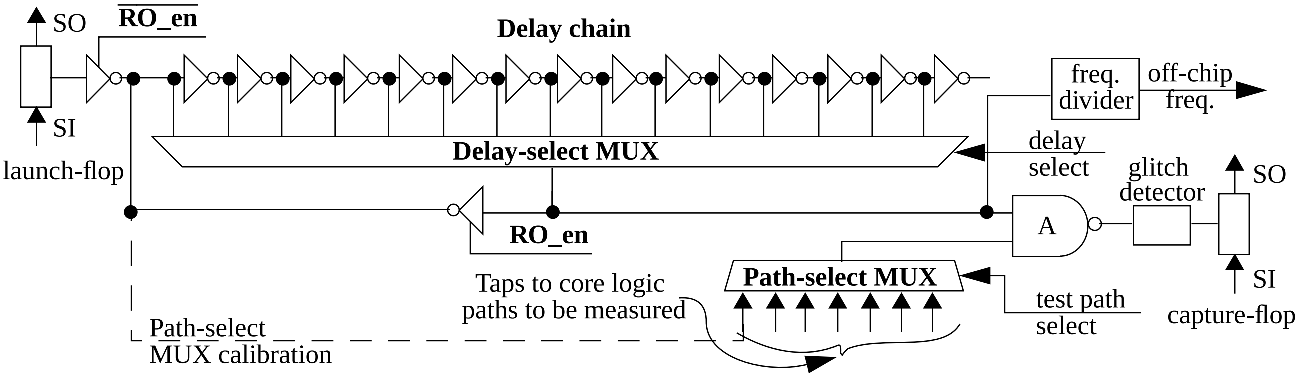

Figure 4. Reference path infrastructure

ing the longest paths maximizes the coverage of small delay defects in the rest of the circuit.

The validation of the reference paths can be carried out in one of two ways. An at-speed delay test can be applied to confirm that each reference path is upper-bounded by the clock cycle time. Although this approach works for the reference paths that are also critical paths in the circuit, it cannot be used to confirm a tight upper timing bound for the shorter reference paths. This is true because the shorter reference paths may have significant slack when tested with an at-speed clock. A straightforward solution for testing the shorter reference paths is to use a faster-than-at-speed clock. Unfortunately, testing at faster-than-at-speed can result in yield loss due to IR drops and other noise effects.

Our approach avoids the drawbacks associated with the application of a faster-than-at-speed clock by incorporating a special reference path on chip. Figure 4 shows the proposed test structure, subsequently referred to as the reference path test structure (**RPTS**). It consists of a launch-flop with scan on the left (to enable standard launch-on-shift/launch-on-capture transition testing), a tri-stateable inverter followed by a string of inverters that form a *delay chain*. The chain is tapped at each successive inverter output using a MUX. The tap point selected determines the delay from the launch-flop to the input of NAND **A** shown on the right in the figure. The MUX *delay select* inputs, which control the selection of the tap point, are controlled by the test engineer using a scan chain.

The NAND **A** gate serves the role of the convergence gate, $\mathbf{G_c}$, described earlier. The bottom input of the NAND A is driven by a *path-select* MUX. The inputs to the MUX are connected to those endpoints that require a faster-than-at-speed validation, as described above. The output of NAND **A** drives a glitch detector, which is identical to the glitch detector described earlier in reference to Figure 3.

The shorter reference paths can be validated by applying a delay test to them while simultaneously launching a transition from the launch-flop of the RPTS. The expected delay of the reference path is emulated in the RPTS by selecting the appropriate tap point in the inverter chain. The result of the race of the transitions along both paths is reflected in output of the RPTS's glitch detector. For example, with the RPTS configured with a delay larger than the reference path under test, the absence of a glitch indicates the reference path under test is shorter and free of small delay defects. The result is recorded in the capture flop for inspection off-chip.

Accurate delay emulation by the RPTS requires knowledge of the actual delay of its delay chain, which can be obtained through calibration. Calibration is performed by configuring the delay chain into a ring oscillator. This is accomplished by setting *RO_en* to 1 in Figure 4. The *delay select* inputs to the MUX are configured so that the entire chain of inverters are part of the ring oscillator. A frequency divider (right side of Figure 4) is used to drive an off-chip pin connected to a frequency measuring instrument. The delay of the chain is simply the inverse of the measured frequency scaled by the value of the frequency divider. Once the RPTS is calibrated, it becomes possible to configure specific delays into the RPTS for validating each of the shorter reference paths.

5.0 ATPG

Successful application of our method requires simultaneous propagation of two transitions, one along a reference path segment and one along a test path segment. To our knowledge, existing ATPG algorithms are not capable of determining tests that meet these requirements. Moreover, once the transitions are propagated to the convergence gate, a sensitized path from the convergence gate to an endpoint must be established. The off-path inputs of this sensitized path segment must be constrained to be glitch free. This is true because any glitching on off-path inputs may invalidate the test.

Our current work is focused on developing an ATPG algorithm designed to meet these constraints. The basic objective is to derive tests for the off-path inputs along each of the reference paths, starting with the off-path inputs closest to the launchpoints and successively working towards the endpoints. This strategy maximizes the opportunity to use validated paths to the off-path nodes as reference paths for tests involving nodes at larger logic depths. Our initial objective is to determine the level of gate delay fault coverage that is attainable on several of the ISCAS'85 and ITC'99 benchmark circuits. Our results will be published in a follow-up paper.

6.0 Summary and Conclusions

In this work, we describe a test method that is able to detect very small delay defects without requiring a faster-than-at-speed clock. The strategy also reduces test power by eliminating the capture clock cycle associated with standard delay testing. The technique makes use of internal races as a means of bounding the delay of one path segment against another. The result of the test either causes a static hazard to be generated or the transition along a reference path to be halted. Glitch detectors are added to path endpoints as a means of distinguishing these two conditions. A reference path test structure is proposed to validate shorter reference paths against small delay defects. This test structure can also be used to aid with correlating models with actual hardware for timing analysis and to measure path delays for validation and debugging of first silicon.

7.0 References

[1] C. Hawkins, A. Keshavarzi and J. Segura, "Parametric Timing Failures and Defect-based Testing in Nanotechnology CMOS Digital ICs", NASA Symposium, 2003.

[2] B. Kruseman, A. K. Majhi, G. Gronthoud and S. Eichenberger, "On hazard-free patterns for fine-delay fault testing," ITC 2004, pp. 213-222.

[3] P. Nigh, "The Increasing Importance of On-line Testing to Ensure High-Reliability Products", ITC, 2003.

[4] P. Gupta and M. S. Hsiao, "ALAPTF: A New Transition Fault Model and the ATPG Algorithm", ITC, 2004, pp. 1053-1060.

[5] W. Qiu, J. Wang, D. M. H. Walker, D. Reddy, X. Lu, Z. Li, W. Shi and H. Balichandran, "K Longest Paths Per Gate (KLPG) Test Generation for Scan-Based Sequential Circuits", ITC, 2004, pp. 223-231.

[6] N. Ahmed, M. Tehranipoor and V. Jayaram, "Timing-Based Delay Test for Screening Small Delay Defects," DAC 2006.

[7] H. Yan, A. D. Singh, "Experiments in Detecting Delay Faults using Multiple Higher Frequency Clocks and Results from Neighboring Die", ITC, 2003.

[8] A. D. Singh and G. Xu, "Output Hazard-Free Transition Tests for Silicon Calibrated Scan Based Delay Testing", VTS, 2006, pp. 349-355.

[9] Cadence Encounter Test, Aug. 2005.

[10] Richard Putman and Rahul Gawde, "Enhanced Timing-Based Transition Delay Testing for Small Delay Defects", VTS, 2006.

[11] B. N. Lee, L. C. Wang and M. S. Abadir, "Reducing Pattern Delay Variations for Screening Frequency Dependent Defects", VTS, 2005, pp. 153-160.